

NPS ARCHIVE
1969
JOHNSON, R.

LARGE-DOMAIN DISCRETE INTEGRATION TECHNIQUES
FOR THE WAVE EQUATION AS AN AID
IN THE CALCULATION OF PROPAGATION LOSS
AND THE STUDY OF ADAPTIVE ACOUSTIC ARRAYS

by

Roy Martin Johnson

United States Naval Postgraduate School



THESIS

LARGE-DOMAIN DISCRETE INTEGRATION TECHNIQUES
FOR THE WAVE EQUATION AS AN AID
IN THE CALCULATION OF PROPAGATION LOSS
AND THE STUDY OF ADAPTIVE ACOUSTIC ARRAYS

by

Roy Martin Johnson

October 1969

T 1315-7

*This document has been approved for public re-
lease and sale; its distribution is unlimited.*

Large-Domain Discrete Integration Techniques for the Wave Equation
as an Aid in the Calculation of Propagation Loss and the Study of
Adaptive Acoustic Arrays

by

Roy Martin Johnson
B.S., University of California, Berkeley, 1954

Submitted in partial fulfillment of the
requirements for the degree of

DOCTOR OF PHILOSOPHY

from the
NAVAL POSTGRADUATE SCHOOL
October 1969

1969

JOHNSON, R.

ABSTRACT

This work is concerned with two inter-related problems: (1) the theory and application of algorithms for the discrete integration of the acoustic wave equation in large, multi-dimensional, variable-parameter domains and, (2) the automatic synthesis of adaptive acoustic arrays for the detection and estimation of received acoustic signals which are incident from such domains.

It is shown that the steady-state integration of large domains may be partitioned into subdomain integrations for slowly-varying variable-parameter domains. A new method of integration based upon the Fast Fourier Transform is given. A new method for obtaining closed form coefficient expressions for the Fast Fourier Transform is shown and illustrated.

The results of a general computer-display simulation program for the Widrow feed-forward algorithm are given. Several limitations and possible modifications to the Widrow procedure are given. The use of Kalman-type filters as an alternative method is introduced.

TABLE OF CONTENTS

| | | |
|------|--|----|
| I. | INTRODUCTION ----- | 15 |
| II. | THE PROPAGATION EQUATIONS ----- | 19 |
| A. | THE UNBOUNDED MEDIUM EQUATIONS ----- | 19 |
| B. | BOUNDARY CONDITIONS ----- | 23 |
| C. | COORDINATE SYSTEMS ----- | 25 |
| D. | THEORETICAL SOLUTIONS TO THE PROPAGATION EQUATIONS ----- | 25 |
| III. | DISCRETE INTEGRATION ----- | 28 |
| A. | DIFFUSION EQUATION ----- | 28 |
| 1. | Theoretical Solution ----- | 28 |
| 2. | Discrete Difference Equations - Instability ----- | 30 |
| 3. | Exact Solution of the Difference Equations ----- | 32 |
| 4. | Refinement of Mesh - Convergence ----- | 24 |
| 5. | Rates of Convergence ----- | 34 |
| B. | IMPLICIT ALGORITHMS AND THEIR SOLUTIONS ----- | 34 |
| C. | DISCRETE MODELS FOR THE WAVE EQUATION ----- | 36 |
| 1. | Continuous - Discrete Integration Approach ----- | 37 |
| 2. | Explicit Scheme ----- | 40 |
| 3. | Solution of Diffusion Equation Implicit System ----- | 42 |
| 4. | Solution of the One-Dimensional Wave Equation Implicit System ----- | 44 |
| 5. | Three-Dimensional Implicit Formulation; Fractional Step Algorithm ----- | 48 |

| | |
|--|----|
| 6. The Fast Fourier Transform Technique For Constant Coefficients ----- | 50 |
| D. BAND-LIMITED PROPERTIES OF DISCRETE MODELS ----- | 59 |
| 1. Cutoff Properties ----- | 60 |
| 2. Dispersion Properties ----- | 61 |
| IV. COMPUTER RESULTS FOR THE MATCHED BOUNDARY CONDITION ----- | 63 |
| A. PARTITIONING OF THE SPACIAL DOMAIN ----- | 63 |
| B. THE INHOMOGENEOUS LINE IMPEDANCE ----- | 64 |
| C. THE MATCHED BOUNDARY CONDITION DOMAIN ----- | 70 |
| 1. Stability of the Internal Energy Criterion ----- | 72 |
| 2. Transient Behavior of Models ----- | 73 |
| 3. Two-Dimensional Model Integration ----- | 74 |
| V. APPROXIMATE FAST FOURIER TRANSFORM COEFFICIENTS ----- | 75 |
| A. INTRODUCTION AND DEFINITION OF TRANSFORM ----- | 75 |
| B. REQUIREMENTS TO OBTAIN COEFFICIENTS BY THE IMPULSE METHOD ----- | 76 |
| C. SPECTRAL REPRESENTATION FOR THE SYMBOLIC IMPULSE FUNCTION ----- | 76 |
| D. DEFINITION OF DISCRETE DERIVATIVE AND ITS SPECTRAL REPRESENTATION ----- | 77 |
| E. ANALYTIC-ENVELOPE EXPRESSIONS FOR DISCRETE FUNCTIONS ----- | 78 |
| F. THEOREM ON DISCRETE DERIVATIVES OF DISCONTINUOUS ANALYTIC-ENVELOPE FUNCTIONS ----- | 78 |
| G. EXAMPLES ----- | 80 |
| 1. Coefficients of a Sawtooth Waveform ----- | 80 |
| 2. Coefficients of Recursive Waveforms - The Exponential Function ----- | 82 |

| | |
|--|-----|
| F. GENERAL PROCEDURE ----- | 85 |
| VI. TWO APPROACHES TO THE OPTIMUM ACOUSTIC ARRAY PROCESSOR ----- | 87 |
| A. INTRODUCTION ----- | 87 |
| B. THE MODIFIED OPTIMUM DETECTION-ESTIMATION ARRAY PROCESSOR ----- | 88 |
| C. ADAPTIVE ARRAYS ----- | 97 |
| 1. The Widrow Scheme ----- | 97 |
| 2. A Visual Display Program ----- | 105 |
| 3. Other Approaches-Future Research ----- | 107 |
| 4. The Use of Feedback Networks; The Kalman Filter -- | 109 |
| VII. CONCLUSIONS ----- | 114 |
| APPENDIX A THE DOUGLAS AND GUNN EQUATIONS ----- | 117 |
| APPENDIX B THE STEP INPUT TO A MATCHED LINE ----- | 119 |
| APPENDIX C THE COSINE INPUT TO A MATCHED LINE ----- | 129 |
| APPENDIX D TWO-DIMENSION VARIABLE-VELOCITY PROBLEM SOLUTION ----- | 146 |
| APPENDIX E TWO-DIMENSION NON-ZERO INITIAL CONDITION SOLUTION ----- | 152 |
| APPENDIX F SOLUTION TO THE WIDROW FILTER ----- | 158 |
| APPENDIX G ILLUSTRATIONS ----- | 207 |
| COMPUTER PROGRAMS ----- | 227 |
| A. TWO-DIMENSIONAL INTEGRATION ROUTINE WV2525 --- | 227 |
| B. THREE-DIMENSIONAL DOMAIN OVERLAY PROGRAM USING RUNGE-KUTTA INTEGRATION ----- | 238 |
| C. RUNGE-KUTTA ADAMS-MOULTON PREDICTOR CORRECTOR WITH DYNAMIC ERROR CONTROL ----- | 277 |
| D. WIDROW FILTER SIMULATION DISPLAY PROGRAM ----- | 277 |

BIBLIOGRAPHY ----- 308

INITIAL DISTRIBUTION LIST ----- 312

FORM DD 1473 ----- 313

LIST OF ILLUSTRATIONS

| | | |
|---------------|--|-----|
| Fig. 1 | Theoretical Solution to the Diffusion Equation ----- | 207 |
| Fig. 2(a,b,c) | Instabilities in the Discrete Diffusion Equation Solution ----- | 208 |
| Fig. 3 | The One-Dimensional Transmission Line Equations ----- | 209 |
| Fig. 4 | Geometric Model of Integration Domains ----- | 210 |
| Fig. 5 | Function Having Discontinuity in Analytic-Envelope at $n=\ell$ ----- | 211 |
| Fig. 6(a) | Function $g(n)=n$ ----- | 211 |
| Fig. 6(b) | Derivative of Function $g(n)=n$ ----- | 211 |
| Fig. 7 | Recursive Example; The Exponential Function ----- | 212 |
| Fig. 8 | Array with Associated Processing Network ----- | 87 |
| Fig. 9 | The Quadratic-Cost-Function Optimal-Estimator Array Processor ----- | 213 |
| Fig. 10 | The General Quadratic-Cost-Function Optimal-Estimator Array Processor ----- | 214 |
| Fig. 11 | General Feedback, Feed-Forward Sampled-Data Network - | 215 |
| Fig. 12 | Wide-Band Feed-Forward Delay Network ----- | 216 |
| Fig. 13 | Narrow-Band Feed-Forward Delay Network ----- | 217 |
| Fig. 14 | The On-Off Adaptive Network Scheme ----- | 218 |
| Fig. 15 | Flow Chart of Array Simulation Program ----- | 219 |
| Fig. 16 | Diagram of Computer Intervention Attention Keyboard ---- | 220 |
| Fig. 17 | Physical Layout of Array Simulation Program ----- | 221 |
| Fig. 18 | Wide-Band Amplitude-Modulated Input Signal ----- | 222 |
| Fig. 19 | Adaption Network With Variable Delays ----- | 223 |

Fig. 20 Mantey's Modified Adaptive Feedback Network ----- 224

Fig. 21 Two-Delay Kalman Filter State Transition Diagram ----- 225

Fig. 22 Two-Delay Kalman Filter Adaptive Network ----- 226

LIST OF SYMBOLS

| <u>Symbol</u> | <u>Description</u> |
|----------------------|---|
| $\kappa(x)$ | Bulk modulus of elasticity |
| ρ | fluid mass density |
| $c(x)$ | speed of sound |
| $p(x,t)$ | pressure |
| t | time |
| x,y,z | cartesian spacial coordinate |
| $\underline{d}(x,t)$ | vector relative particle displacement |
| $\underline{v}(x,t)$ | vector velocity |
| α | normalization constant; attenuation coustant |
| β | normalization constant; phase constant |
| \underline{n} | normal unit vector |
| $G(f)$ | signal frequency spectrum |
| B | signal spectrum bandwidth |
| α | heat capacity |
| η | heat conductivity |
| T_m^n | temperature at position $m\Delta x$ at time $n\Delta t$ |
| τ | discrete expansion time growth constant |
| Δt | discrete time increment |
| Δx | spacial grid increment |
| ζ_x | stability factor for explicit algorithm prediction |

| | |
|---------------------|---|
| H_1 | Fourier Transform prediction stability matrix |
| G | implicit integration scheme stability gain matrix |
| \underline{u}_m^o | implicit integration scheme state vector |
| \hat{p}_k^o | FFT pressure wavenumber coefficients |
| G_{10} | implicit scheme FFT wavenumber gain matrix |
| $\delta_{k,\ell}$ | Kronecker delta function |
| \underline{u}_h | homogeneous solution state vector |
| \underline{u}_i | inhomogeneous solution state vector |
| γ | complex propagation constant |
| ω | angular frequency |
| f_c | discrete-model cutoff frequency |
| z_o | characteristic wave impedance |
| $z(x)$ | complex wave impedance |
| R | series resistance / length of transmission line |
| L | series inductance / length of transmission line |
| G | shunt conductance / length of transmission line |
| C | shunt capacitance / length of transmission line |
| z_s | series impedance / length of transmission line |
| Y_{sh} | shunt admittance / length of transmission line |
| $R(x)$ | voltage reflection coefficient |
| $\hat{P}(x)$ | pressure phasor |
| $\hat{V}(x)$ | velocity phasor |
| T_{micro} | micro domain kinetic energy |

| | |
|-------------------------------------|---|
| V_{micro} | micro domain potential energy |
| U_{macro} | total macro domain internal energy |
| $G(k)$ | discrete signal spectrum |
| $g(n)$ | discrete time signal |
| $\delta_N(s-\ell)$ | periodic Kronecker delta |
| $Us_N(s-\ell)$ | periodic Heaviside function |
| $\underline{x}(t)$ | signal input vector |
| $\underline{y}(t)$ | signal output vector |
| $\underline{m}(t)$ | observation transformation vector |
| $\underline{x}'(k)$ | discrete input signal spectrum vector |
| $\sigma'(k)$ | discrete desired input signal spectrum |
| $\underline{\mu}'(k)$ | discrete observation transformation spectrum vector |
| $\underline{n}(t)$ | input noise excitation vector |
| $\underline{\eta}'(k)$ | input noise spectrum vector |
| $\underline{R}(k)$ | input noise unto-correlation vector |
| $\underline{P}(k)$ | input power spectral density vector |
| $\zeta(T)$ | test statistic for detection problem |
| $p(\underline{\psi}/\underline{n})$ | conditional probability of input signal |
| $\underline{\theta}(k)$ | "pre-whitening" spectral vector |
| $\Omega(k)$ | spectral output response |
| λ | wavelength |
| a_i | feedback coefficients |
| b_i | feed-forward coefficients |

| | |
|--------------------|---|
| z | Z-transform complex variable representing unit delay |
| $\underline{x}(z)$ | Z-transform of array output vector |
| $G_i(z)$ | Z-transform transfer function for i-th array element. |
| $\underline{w}(t)$ | feed-forward weight vector |
| $\epsilon(k)$ | discrete quadratic error signal |
| $H(k)$ | observation matrix |
| $\Delta(k, k-1)$ | excitation noise transition matrix |
| $\Phi(k, k-1)$ | state transition matrix |
| $G(k)$ | Kalman filter gain matrix |
| $\hat{s}(k/k)$ | best conditional estimate of desired input signal given k previous estimates |

ACKNOWLEDGEMENT

The author wishes to acknowledge the sincere and untiring efforts of Dr. Harold A. Titus in the guidance and direction of this research.

The author also wishes to echo with humbleness the Prayer of Thanksgiving by R. C. K. Lee:

"Before God, all men are equally foolish,
for we cannot begin to comprehend all the
infinite mysteries surrounding us. If there
be a grain of wisdom in this work, may it
be to His glory, for He alone hath provided
me with this splended opportunity for learn-
ing, the wonderful parents, teachers and
friends, a harmonious home, excellent
health -- and most of all, for the strength
ever-sufficient from day to day."

I. INTRODUCTION

The detection, estimation, and distribution of acoustic energy signals propagating in inhomogeneous seawater are subjects of continuing interest. These subjects reach a focus of inter-relation in the design of efficient acoustic receiving arrays; such arrays should match the propagation solution properties of the location where the receiver is to be placed.

Past investigations into the trajectory and strength of acoustic signals in seawater have relied almost totally upon minimum-time plane-wave ray-path techniques or approximate normal mode solutions to crude stratification models. The theoretical and computational economy of these methods is their main asset. Rarely is a single model sufficient to represent faithfully the propagation characteristics of long acoustic paths involving multi-dimensional energy spreading or inhomogeneous medium phenomena such as convergence zones.

A more fundamental approach to such problems is the direct integration of the appropriate variable-coefficient wave partial-differential equation with any receiving array present being treated as part of boundary constraints. Unfortunately, no general theoretical methods exist for treating variable-coefficient partial-differential equations. However, foreseeable order of magnitude increases in digital computation speed and memory capacity make the direct finite-difference numerical integration of the wave equation a distinct possibility.

This paper attempts to make contributions to both the numerical integration techniques relative to solution of the large-domain variable-parameter acoustic wave equation and the theory and practice of adaptive acoustic antenna arrays.

In chapter II the acoustic wave partial-differential equation with variable coefficients is derived in properly posed form from first principles assuming infinitesimal deformations and that the coefficients are slowly varying and time independent. The appropriate boundary conditions are also discussed with particular emphasis being devoted to the matched boundary condition.

In chapter III methods for finite difference integration of discrete models for partial-differential equations are introduced. The problems of instability and convergence are illustrated for the one-dimensional diffusion equation which is the simplest partial-differential equation of common interest which has been extensively studied.

Continuous-discrete, explicit, and implicit solution algorithms are then studied and the results of actual computer calculations relative to required computation time, accuracy and stability are presented and discussed. Along with these results, a new method for semi-implicit integration of the three-dimensional wave equation using the Fast Fourier Transform algorithm of Cooley and Tukey is given. The method appears faster than all other methods tested with the exception of the successive approximation tri-diagonal method of Douglas and Gunn.

The magnitudes of the spacial increments Δx , Δy , Δz for proper wave equation simulations are obviously highly important; it is shown that the highest desired propagation frequency places a maximum value on these increments which is considerably less than the appropriate Shannon restriction.

The problem of integrating much larger spacial domains than could possibly be contained in any reasonable computer memory is taken up in chapter IV. General transient problem integrations appear to be very difficult to treat in such cases; the breaking up of the large integration domain into smaller memory requirement domains which can be integrated is frustrated in the variable parameter case by the question of which small domain is to be integrated first. It is shown however, that the steady state solution problem can be treated in such a manner for the slowly-varying parameter case by making use of the matched impedance boundary condition of ordinary transmission line theory. Since the problem of calculating steady state acoustic propagation loss is of this type the result is significant. The chapter concludes by presenting the results of several computer wave-equation integration problems.

The Fast Fourier Transform algorithm is a very efficient digital computer algorithm for computing the discrete Fourier transform of a finite-length sampled input function. Although a few discrete finite transforms can be obtained in closed form by use of the geometric sum formula, no general techniques for obtaining the explicit closed-form coefficients presently are known. In chapter V a new technique for obtaining such forms is given and two examples are worked as illustrations. The method is based on

taking first-order backwards finite differences and is quite general.

The subject of acoustic detection and estimation arrays is taken up in chapter VI. From the aspect that the local medium propagation properties may vary with the time and as a consequence the most efficient detector should also, the concept of automatic self-adapting arrays appear especially appealing. Therefore, after initially considering arrays from the classical optimum detection point of view, adaptive arrays are studied. A quite sophisticated general purpose interactive-display digital computer program has been used to study several possible automatic synthesis adaptive antenna arrays which may be placed on the boundary of a large wave-equation integration domain. General results are discussed and particular emphasis is devoted to possible modifications of existing adaption algorithms in order to increase their adaption speed.

II. THE PROPAGATION EQUATIONS

A. THE UNBOUNDED MEDIUM EQUATIONS

The problem of propagation of acoustic waves in salt water will be assumed to satisfy an acoustic wave equation; however, the solution for domains with variable parameters in more than one dimension is very complicated in general so that it is advisable to begin the analysis with a derivation of the medium equations from first principals in order to make clear the assumptions involved.

For infinitesimal acoustic disturbances in fluids the volume bulk modulus κ is related to the sound speed c by the relation^[1]

$$\kappa = \rho c^2 \quad (1)$$

where ρ is the density of the fluid. For the range of conditions dealt with here, the density ρ will be considered to be a constant. The acoustic speed c , however, is a function of both temperature and pressure.^[2]

Notwithstanding these complications, it is usual to assume that the resulting equation of state

$$c = c(p, t) \quad (2)$$

can be reduced to a parametric, single-valued function of position

$$\begin{aligned} c &= c(x, y, z) \\ &= c(\underline{x}) \end{aligned} \quad (3)$$

which is constant with respect to time for periods long compared with the periods of oscillations of interest.^[2] By Hookes law,^[1] the compression of particles relative to an increment change in pressure, $p(x, y, z, t)$ is

$$p(x, y, z, t) = -\kappa \nabla \cdot \underline{d}(x, y, z, t) \quad (4)$$

where $\underline{d}(\underline{x},t)$ is the vector relative displacement of the particles. Taking the partial derivative of (4) with respect to time gives

$$\frac{\partial p}{\partial t} = - \kappa \nabla \cdot \underline{v}(\underline{x},t) \quad (5)$$

where $\underline{v}(\underline{x},t)$ is the vector acoustic disturbance velocity. The two dependent variables p, \underline{v} are further related by Newton's second law which relates the vector acceleration of a sample mass volume to the pressure gradient

$$\rho \frac{d\underline{v}}{dt} = - \nabla p. \quad (6)$$

It is usual to further simplify (6) by assuming that in the total velocity derivative

$$\frac{d\underline{v}}{dt} = \frac{\partial \underline{v}}{\partial t} + (\nabla \cdot \underline{v}) \underline{v} \quad (7)$$

the second term is of second order for all but the very highest possible frequencies so that

$$\rho \frac{\partial \underline{v}}{\partial t} = - \nabla p. \quad (8)$$

The two equations (5) and (8) constitute the fundamental first-order partial-differential equations to be solved in the absence of dissipative losses. The effects of dissipative losses usually are included empirically after solution of the lossless equations (5) and (8).^[1]

Equation (8) is separable into three scalar partial-differential equations so that with (5) there are a total of four first-order partial differential equations to be solved:

$$\frac{\partial p}{\partial t} = - \kappa \left(\frac{\partial v_x}{\partial x} + \frac{\partial v_y}{\partial y} + \frac{\partial v_z}{\partial z} \right) \quad (9)$$

$$\frac{\partial v_x}{\partial t} = - \frac{1}{\rho} \frac{\partial p}{\partial x} \quad (10)$$

$$\frac{\partial v_y}{\partial t} = - \frac{1}{\rho} \frac{\partial p}{\partial y} \quad (11)$$

$$\frac{\partial v_z}{\partial t} = - \frac{1}{\rho} \frac{\partial p}{\partial z} \quad (12)$$

The bulk modulus κ is a function of position since by (3) c is a function of position and ρ has been assumed constant. The set of equations (9, 10, 11, 12) are a mixed set of equations in that they are functions of two dependent variables p and \underline{v} . They may be unmixed by taking the partial of (9) with respect to t ,

$$\frac{\partial^2 p}{\partial t^2} = - \kappa \left(\frac{\partial}{\partial t} \left(\frac{\partial v_x}{\partial x} \right) + \frac{\partial}{\partial t} \left(\frac{\partial v_y}{\partial y} \right) + \frac{\partial}{\partial t} \left(\frac{\partial v_z}{\partial z} \right) \right) \quad (13)$$

which assuming continuity of first derivatives may be written as

$$\begin{aligned} \frac{\partial^2 p}{\partial t^2} &= - \kappa \nabla \cdot \left(\frac{1}{\rho} \frac{\partial \underline{v}}{\partial t} \right) \\ &= \frac{\kappa}{\rho} \nabla \cdot \nabla p \\ &= c^2 \nabla^2 p \end{aligned} \quad (14)$$

This is a scalar wave equation for the pressure. The velocity \underline{v} satisfies an associated type of equation; if the partial of (8) with respect to t is taken and continuity is again assumed,

$$\begin{aligned} \frac{\partial^2 \underline{v}}{\partial t^2} &= - \frac{1}{\rho} \nabla \left(\frac{\partial p}{\partial t} \right) \\ &= - \frac{1}{\rho} \nabla (- \kappa \nabla \cdot \underline{v}) \\ &= \frac{\kappa}{\rho} \nabla (\nabla \cdot \underline{v}) \nabla \kappa \end{aligned} \quad (15)$$

The problems of interest here will be confined to unbounded fluids which are assumed unable to support shear stress waves; such being the case the curl of \underline{v} must vanish and hence by the vector identity

$$\nabla (\nabla \cdot \underline{v}) = \nabla \times (\nabla \times \underline{v}) + \nabla^2 \underline{v} \quad (16)$$

(15) becomes

$$\frac{\partial^2 \underline{v}}{\partial t^2} = \frac{\kappa}{\rho} \nabla^2 \underline{v} + \frac{1}{\rho} (\nabla \cdot \underline{v}) \nabla \kappa. \quad (17)$$

The second term in (17) is usually neglected as being of second order since $\nabla \cdot \underline{v}$ is assumed small for the reason previously stated and in addition $\nabla \kappa$ can be made also small by proper space partitioning in computer analyses. In ray-path solutions the second term can become important in certain cases. [5]

As previously stated, the equations (9,10,11,12) are fundamental; the derived set (14) and (17) are no less valid but their domain of solutions is larger. [4] It is possible for certain pathological solutions to exist satisfying (14) and (17) under a given set of boundary conditions but these solutions none-the-less will not satisfy (9,10,11,12). Thus only (9,10,11,12) are both necessary and sufficient. The pathological solutions to (14) and (17) will almost never occur in practice if (14) and (17) are solved analytically since the investigator will invariably pick characteristic forms in separation procedures which are physically regular in behavior; the problem is not academic however in computer solutions of the equations since poorly defined (real world) boundary conditions could possibly lead to successive approximate solutions converging to these pathological solutions. For all of these reasons the integration techniques in this work

will apply to the equations (9,10,11,12).

The set of equations (9,10,11,12) is also properly posed, i.e., a norm can be defined in an appropriate Banach space which is bounded.^[4]

In implicit algorithm solutions to (9,10,11,12), it is very convenient to introduce new variables to make the equations (5) and (8) completely symmetric. Let

$$\underline{v} = \alpha \underline{v}' \quad (18)$$

$$p = \beta p' \quad (19)$$

then to first order

$$\frac{\partial p'}{\partial t} = -c \nabla \cdot \underline{v}' \quad (20)$$

$$\frac{\partial \underline{v}'}{\partial t} = -c \nabla \cdot p' \quad (21)$$

where

$$\begin{aligned} \beta &= 1 \\ &= \text{constant} \end{aligned} \quad (22)$$

$$\alpha(\underline{x}) = \frac{1}{\rho c(\underline{x})} \quad (23)$$

The first order assumption in (20) and (21) assumes that $\alpha(\underline{x})$ is a slowly varying function of position. In addition to symmetry the new variables have a much reduced dynamic range which considerably increases the accuracy of computer computations.

B. BOUNDARY CONDITIONS

The equations (5) and (8) require a complete, consistent set of boundary conditions in order to guarantee that their solution will be unique and properly posed. The usual discrete methods of solution, from a practical if not a mathematical point of view, almost always are limited to bounded

domains. This excludes several important practical problems of interest including wave propagation in semi-infinite media. For the special case in which the steady-state propagation properties are desired it will be shown that by an appropriate spacial partitioning of the volume domain of interest together with an extension of the matched boundary condition of one-dimensional transmission line theory this problem may be solved.

The boundary condition at a fixed wall requires that the normal component of velocity vanish

$$\underline{n} \cdot \underline{v}(\text{bnd}, t) = 0 \quad (24)$$

where \underline{n} is a unit normal to the boundary surface.

An equivalent, not independent, condition on the pressure is from (6)

$$\underline{n} \cdot \nabla p|_{\text{bnd}} = 0. \quad (25)$$

At a free surface, if surface tension forces and atmospheric pressure may be ignored, the pressure must vanish

$$p(\text{surface}, t) = 0. \quad (26)$$

The matched boundary condition relates the pressure and velocity at a point and is an example of a mixed boundary condition. The condition is most fundamentally applied to one dimensional problems and is precisely reflectionless for constant parameter applications. The condition is

$$\frac{p}{|\underline{v}|} = \sqrt{\kappa\rho} \quad (27)$$

where $|\underline{v}|$ represents the magnitude of the velocity. The condition can be approximately extended to variable-parameter media as well as multiple dimensions.

C. COORDINATE SYSTEMS

The equations (5) and (8) are in vectorial form even though derived from rectangular coordinate considerations; they hence hold true for any orthogonal curvilinear coordinate system. The choice of coordinate system depends upon the type of problems to be investigated as well as the domain size and approximations to be utilized. For example, physical problems inherently possess propagation properties which are three-dimensional, but frequency, source location, parameter variations and fixed boundaries may allow the propagation process to be confined to only one or two dimensions.

The discussions in this work are confined to rectangular and cylindrical coordinates but the technique often can be extended in a simple manner to spherical coordinates. The theory is usually illustrated for simplicity in one dimension but the computer simulations that have been carried out are always at least two-dimensional.

D. THEORETICAL SOLUTIONS TO THE PROPAGATION EQUATIONS

To measure the accuracy of computer solutions to (5-8), theoretical solutions are desirable. In those cases where the parameters are constants, the ordinary theoretical acoustic solutions to the wave equation in homogeneous media suffice; in the general case of inhomogeneous media exact theoretical solutions are almost non-existent except for some very special cases. The case of linear periodic-parameter media has received extensive treatment^[5] and constitutes the most useful theoretical solution known as a comparison standard for finite discrete models of the wave

equation. In chapter III it is shown that such models are inherently dispersive, whereas the true wave equation in unbounded media is perfectly non-dispersive. However, if the partitioning is sufficiently small, the low frequency or Rayleigh limit approaches precisely the non-dispersive limit.

The step function is used as a test function in this work to verify that the Rayleigh limiting velocity is approached and that dispersive waveform distortion is not prohibitive.

Finite difference approximations to the wave equation are band-limited inherently so that the infinite slope at the discontinuity of a step function will never be represented exactly in computer solutions. The maximum attainable slope of the solution is most easily found by use of the Fourier transform.^[6] Assume that the resulting solution at some arbitrary point \underline{a} and time t is band-limited to $|f| < B$ hertz. Then by the Fourier transform the resulting pressure is represented by

$$p(\underline{a}, t) = \int_{-B}^B G(\underline{a}, f) e^{j2\pi ft} df \quad (28)$$

where $G(\underline{a}, f)$ is the spectral resolution function. Hence

$$\left| \frac{\partial p}{\partial t} \right| \leq 2\pi B \cdot \text{Max}(|G(\underline{a}, f)|) \cdot 2B. \quad (29)$$

Assuming that the pressure wave is an outward-going wave in the x direction only then the velocity in the x direction, by (27) is directly proportional to the pressure and hence by (9)

$$\begin{aligned} \frac{\partial p}{\partial t} &= -\kappa \frac{\partial v_x}{\partial x} \\ &= -\kappa \frac{1}{\sqrt{\kappa \rho}} \frac{\partial p}{\partial x} \end{aligned} \quad (30)$$

or

$$\begin{aligned}
 \left| \frac{\partial p}{\partial t} \right| &= \sqrt{\frac{\rho}{\chi}} \left| \frac{\partial p}{\partial t} \right| \\
 &\leq \sqrt{\frac{\rho}{\chi}} \cdot 4\pi B^2 \text{Max} (|G(\underline{a}, f)|) .
 \end{aligned}
 \tag{31}$$

Thus the frequency band-limitedness of discrete models of the wave equation has as a direct consequence the result that the spacial representation of waveforms is also spacial-slope limited. In essence, then, transient-solution curves for step-function time input to discrete models of the wave equation will exhibit finite-maximum spacial-slope properties as the solution is observed at various fixed times. This has been observed in computer solutions and will be discussed in the chapter on discrete models for the wave equation.

III. DISCRETE INTEGRATION

A. THE DIFFUSION EQUATION

The purpose of this chapter is to introduce discrete integration models for partial differential equations suitable for programming on digital, analog, or hybrid computing machines. The relevant theory of such models is extensive. The heat diffusion equation in one space dimension is the simplest partial-differential equation possessing known theoretical solutions and which also illustrates the problems of instability and convergence common to discrete difference methods of solution. The wave equation, being a hyperbolic partial differential equation of the second order, is inherently more difficult to solve than the diffusion equation which is parabolic and of first order so that the relevant theory will be illustrated for the diffusion equation with necessary modifications for the wave equation being taken up in section C.

1. Theoretical Solution

The diffusion of heat through a one-dimensional solid of thermal conductivity η and heat capacity α is given by the parabolic partial differential equation

$$\alpha \frac{\partial T}{\partial t} = \frac{\partial}{\partial x} \left(\eta \frac{\partial T}{\partial x} \right) \quad (32)$$

where $T(x,t)$ is the temperature of the solid at position x and time t . Both the heat capacity and conductivity can in general be functions of position and time; it shall be assumed here that they are constants so that (32) becomes

$$\frac{\partial T}{\partial t} = \sigma \frac{\partial^2 T}{\partial x^2} \quad (33)$$

where

$$\sigma = \frac{\eta}{\alpha} . \quad (34)$$

For a domain assume the real line $0 \leq x \leq \pi$ and that $t \geq 0$.

Further assume the given boundary values

$$T(x, 0) = \varphi(x) \quad (35)$$

and

$$T(0, t) = T(\pi, t) = 0. \quad (36)$$

As an example suppose

$$\begin{aligned} \varphi(x) &= cx, \quad 0 \leq x \leq \frac{\pi}{2} \\ &= c(\pi - x), \quad \frac{\pi}{2} \leq x \leq \pi . \end{aligned} \quad (37)$$

The theoretical solution to this boundary value problem is obtainable in convergent infinite Fourier series form; the assumed solution

$$T(x, t) = \sum_{m=-\infty}^{\infty} A_m(t) e^{jmx} \quad (38)$$

upon substitution in (33) requires that

$$\sum_{m=-\infty}^{\infty} \left\{ \frac{dA_m}{dt} - \sigma (jm)^2 A_m \right\} e^{jmx} = 0 \quad (39)$$

which can only be true for arbitrary values of x provided

$$\frac{dA_m}{dt} + m^2 \sigma A_m = 0 . \quad (40)$$

The solution to (40) which is bounded for all $t \geq 0$ is

$$A_m(t) = A_m(0) e^{-\sigma m^2 t} . \quad (41)$$

Hence (38) may be written as

$$T(x, t) = \sum_{m=-\infty}^{\infty} A_m(0) e^{jmx - m^2 \sigma t} \quad (42)$$

The coefficients $A_m(0)$ are obtained by the usual coefficient formula from the distribution of temperature at $t = 0$:

$$\begin{aligned} A_m(0) &= \int_{-\pi}^{\pi} \varphi(x) e^{-jmx} dx \\ &= 0, \quad m \text{ even} \\ &= \frac{2jC}{m^2 \pi}, \quad m \text{ odd}. \end{aligned} \quad (43)$$

This theoretical solution has been evaluated for different values of the time by summing the series (42) with coefficients (43) on an IBM 360/67 to six decimal places; the solution is shown in Fig. 1.

The use of the Fourier series method of solution is limited strictly to linear constant-coefficient partial-differential equations whereas the finite difference methods to follow can be applied to much more general types of problems; the theoretical solution however does give a standard of comparison for any proposed finite-difference integration scheme when such a scheme is applied to linear constant-coefficient problems.

2. Discrete-Difference Equation - Instability

A great number of possible discrete models can be proposed to integrate numerically the equation (33) with the domain of (35,36). The majority of methods discretize x and t into some kind of grid space in order to convert the solution of the partial-differential equations or a set of algebraic equations. The grid may be uniform or non-uniform in the coordinates $m\Delta x, n\Delta t$; the crucial questions to be answered for any scheme, however, usually are as follows:

- A. Is the integration scheme stable as the number of points in the $m\Delta x, n\Delta t$ solution space become infinite?
- B. Does successive refinement of the solution by increasing the number of grid points in the solution space produce a solution which converges uniformly to the theoretical solution?

C. From a practical point of view the number of grid points must always remain finite. At what rate as $\Delta x, \Delta t \rightarrow 0$ does the error between the finite difference approximation and the exact solution tend to zero?

The answers to these questions are exceedingly difficult in general; usually each case must be treated separately, especially in variable coefficient or non-linear problems. The known theory relative to questions A and B is fairly complete, especially for linear problems; the present status of knowledge about question C is quite inadequate since relative few exact solutions are known for variable coefficient problems.

The simplest and most commonly applied discrete-approximation scheme to the problem solved above is to discretize the space variable x into M sub-intervals Δx where

$$\Delta x = \frac{\pi}{M} , \quad (44)$$

and then to successively iterate forward in time the $M-1$ equations

$$\frac{T_m^{n+1} - T_m^n}{\Delta t} = \frac{\sigma}{(\Delta t)^2} \left[T_{m+1}^n - 2T_m^n + T_{m-1}^n \right] , \quad (45)$$

$$m = 1, \dots, M-1, n = 0, 1, 2, \dots$$

The boundary conditions given in (36) become

$$\begin{aligned} T_0^n &= T(0, n \Delta t) \\ &= 0 \quad , n = 0, 1, \dots, \end{aligned} \quad (46)$$

$$\begin{aligned} T_M^n &= T(M \Delta x, n \Delta t) \\ &= 0 \quad , n = 0, 1, \dots \end{aligned} \quad (47)$$

and the initial condition (37) prescribes

$$\begin{aligned} T_m^0 &= T(m \Delta x, 0) \\ &= \varphi(m \Delta x) \quad , \quad 0 \leq m \leq M \end{aligned} \quad (48)$$

Nothing has been said about the magnitude of the time increment Δt ; this is the crux of the problem as is indicated in the plots of the computer solution to (45) shown in Figs. 1, 2(a,b,c). For small values of Δt the solution predicted by (43) agrees quite well with the theoretical solution shown in Fig. 1; when the value of Δt becomes large the computed solution becomes unstable as shown in Fig. 2(a,b,c). This is not caused by roundoff or truncation errors of the digital computer (double precision calculations have no effect) but is a direct result of the instabilities inherent in (45).

3. Exact Solution of the Difference Equations

Von Neumann^[36] was the first investigator to successfully explain the instability inherent in the system (45) by means of Fourier analysis. To apply the method to (45) assume a series of the form

$$T_m^n = \sum_{k=-\infty}^{\infty} B_k \tau^n e^{jk(m \Delta x)} \quad (49)$$

is substituted in (45). After some algebra it can be seen that (49) satisfies the system provided

$$\tau(k) = 1 - \frac{2\sigma \Delta t}{(\Delta x)^2} (1 - \cos(k \Delta x)) \quad (50)$$

Further, if the coefficients B_k are set equal to the corresponding $A_m(o)$ in (43), the solution (49) is exactly the same as the previous Fourier solution (42) except for the time dependence function. In (49) only the values at the actual mesh points $m \Delta x$, $n \Delta t$ of course have meaning. Assuming then, that both the Fourier solution (42) and the solution

$$T_m^n = \sum_{k=-\infty}^{\infty} A_k(o) \tau^n e^{jk(m \Delta x)} \quad (51)$$

are to be evaluated at the mesh points $m \Delta x$, $m = 0, 1, \dots, M$, the essential difference in the solutions occurs in the amplitude of the time factors $(\tau(k))^n$ and $e^{-\sigma m^2 t}$. The factors represent the time dependent performance of the individual Fourier harmonics in respective solutions and because of linearity are independent of each other. The exponential factor $e^{-\sigma m^2 t}$ clearly decays for increasing time whereas the factor

$$(\tau(k))^n = (1 - k^2 \sigma \Delta t + \frac{1}{12} k^4 \sigma \Delta t (\Delta x)^2 - \dots)^n \quad (52)$$

tends to zero for increasing n only if

$$|\tau(k)| \leq 1. \quad (53)$$

If for any harmonic the inequality (53) is not satisfied, that particular harmonic will grow as time increases leading to the instability shown in Fig. 2(a,b,c).

From (50) τ is real for all values of k and never more positive than +1. The only possibility for instability thus must result from those cases where τ is less than -1. The worst condition occurs for the particular value of $k = k_c$ such that

$$\cos(k_c \Delta x) = -1, \quad (54)$$

which for stability restricts

$$\frac{2\sigma \Delta t}{(\Delta x)^2} \leq 1. \quad (55)$$

Conceptually, the equality in (54) occurs for those harmonics of wave-number k_c whose wave-lengths are of the order of magnitude of twice the discrete spacial grid spacing Δx . If these harmonics have amplitude exceeding the inequality restriction in (52) they will become unacceptably large as the time progresses. It has been shown that the inequality (55) is both necessary and sufficient for stability of this discrete integration scheme.

4. Refinement of Mesh - Convergence

If the domain of x does not change, it is clear that the magnitude of Δx will decrease as M increases, i.e., as the grid or mesh is refined. Question B previously stated in essence asks whether, as the grid is refined the limit

$$\lim_{M \rightarrow \infty} |T(m\Delta x, n\Delta t) - T_m^n| \rightarrow 0 \quad (56)$$

holds true. In (56) $T(m\Delta x, n\Delta t)$ is the value of the temperature calculated by substituting the values $m\Delta x, n\Delta t$ in the exact theoretical solution of the diffusion equation. Hildebrand^[37] has proven that the restriction (55) is sufficient to guarantee the limit (56). Further it can be stated that if a like analysis can be performed on other discretization algorithms for solution of the diffusion equation which lead to analogous expressions equivalent to (55) the sufficiency of convergence follows.

5. Rates of Convergence

The problem of rate of convergence (question C), usually must be handled on an individual basis and experience in actual computation is still the fundamental requirement.

In the case of the wave equation it will be shown in Chapter III that the minimum number of spacial increments required is set by the upper frequency limit for which the propagation properties are desired and by the characteristic acoustic velocity of the medium.

B. IMPLICIT ALGORITHMS AND THEIR SOLUTION

The restriction (55) on the relationship between Δt and Δx for stability is very severe; if Δx is refined by a factor of $\frac{1}{2}$, Δt must be

reduced to $\frac{1}{4}$ of its previous value to guarantee stability. The solution algorithm (45) is called an explicit algorithm in that it explicitly predicts T_m^{n+1} directly. Implicit solution algorithms are also available which require the solution of a set of simultaneous equations in order to find T_m^{n+1} . The more involved computations necessary are more than offset by the fact that the stability criterion may be much less severe, or in some cases, the system of equations may be unconditionally stable. In the latter cases the magnitude of the increment Δt is independent of Δx and can be set purely on the basis of the computing machine truncation error involved or the fineness of the final solution grid desired.

In general, if the partial differential equation is linear and only nearest-neighbor interactions are involved in the corresponding explicit algorithm, the choice of an implicit algorithm is indicated since the solution of the resulting linear equations involves inverting a nearly diagonal matrix.

Reference 4 lists a great number of implicit schemes for the diffusion equation, some of which are unconditionally stable. A typical example of an implicit scheme is the equation

$$\frac{T_m^{n+1} - T_m^n}{\Delta t} = \frac{\sigma}{(\Delta x)^2} \left[\beta (T_{m+1}^{n+1} - 2T_m^{n+1} + T_{m-1}^{n+1}) + (1-\beta) (T_{m+1}^n - 2T_m^n + T_{m-1}^n) \right] \quad (57)$$

where β is an adjustable parameter. If $\beta = 0$ the scheme reduces to the explicit scheme considered previously; if $\beta = \frac{1}{2}$ a centered time-difference equation results which is implicit. The equation (57) can be investigated as previously for stability; the condition equivalent to (55) is

$$\frac{2\sigma \Delta t}{(\Delta x)^2} \leq \frac{1}{1-2\beta} , \quad \text{if } 0 \leq \beta < \frac{1}{2} \quad (58)$$

no restriction , if $\frac{1}{2} \leq \beta \leq 1$.

Hence, for example if $\beta = \frac{1}{2}$ the equation is unconditionally stable.^[7]

It can be written in the form

$$\begin{aligned} -\zeta T_{m+1}^{n+1} + (2\zeta + \frac{1}{\Delta t}) T_m^{n+1} - \zeta T_{m-1}^{n+1} \\ = (\frac{1}{\Delta t} - 2\zeta) T_m^n + \zeta T_{m+1}^n + \zeta T_{m-1}^n \end{aligned} \quad (59)$$

where

$$\zeta = \frac{\sigma}{2(\Delta x)^2} . \quad (60)$$

The equation (59) contains M-1 unknowns, i.e., all of the variables on the left side of (59) minus the two known boundary conditions supplied by (46) and (47). The right side of (59) contains all known values so that the M-1 equations are sufficient for solution of the system by any of the standard methods applicable to the simultaneous solution of linear equations.

The equations (59) can be written in the general form

$$A \underline{T}^{n+1} = B \underline{T}^n \quad (61)$$

in which A and B are tri-diagonal matrices. For the one-dimensional case there exists a particularly efficient adaption of the Gauss elimination algorithm for the solution of this system which will be discussed in section C on the wave equation.

C. DISCRETE MODELS FOR THE WAVE EQUATION

The wave equation, in component symmetrical form given by (20) and (21), is inherently more difficult to solve than the diffusion equation considered in section A. To begin with it is of second order in the time

as well as in space and hence requires twice as many boundary values and at least twice as much computation to solve. Further it is a hyperbolic equation and the solutions need not be analytic for all values of the independent variables as is the case always for the diffusion equation. [39] The formulation (20) and (21) constitute a properly posed [4] system in that p' and \underline{v}' constitute components of a general vector \underline{u} which is dense and possesses finite norm in the appropriate Banach space. [8]

1. Continuous - Discrete Integration Approach

The simplest conceptual model for integrating the wave equation numerically is obtained by approximating the partial spacial derivatives occurring in (20) and (21) by finite differences. Although the approach is limited in digital applications by very severe restrictions on the time increment Δt in order that stability be maintained, the approach is fundamentally valid and easy to apply being successfully used by many investigators. To be efficient computationally the procedure requires a digital computer having a great deal of parallel processing capability or an analog type computer. A general hybrid machine of large capacity would probably be ideal but as yet the amount of operational amplifiers and electronic switches required have prohibited the economic construction of such a machine. With the advent in the near future of integrated circuit technology this approach may become dominant.

The result of replacing the partial spacial derivatives by finite difference approximations at a set of spacial grid points is to produce a set of ordinary-differential equations for the pressure and velocity at these grid points. To illustrate the method assume in (20) and (21)

that the propagation is in the x direction with no variations in the y and z directions. The one-dimensional first-order equations in normalized form are then

$$\frac{\partial p}{\partial t} = -c \frac{\partial v_x}{\partial x} \quad (62)$$

$$\frac{\partial v_x}{\partial t} = -c \frac{\partial p}{\partial x} \quad (63)$$

where the superscripts have been dropped. The equations are completely analogous to the electrical transmission line equations^[38] in which pressure is analogous to signal voltage and velocity is analogous to signal current. (In the un-normalized form of (62) the coefficient c would be replaced by the reciprocal of the series inductance/length.)

Conceptually the single variables p , v_x are replaced by column vectors \underline{p} and \underline{v}

$$\underline{p}(t) = \begin{pmatrix} p_0(t) \\ \vdots \\ p_M(t) \end{pmatrix} \quad (64)$$

$$\underline{v}(t) = \begin{pmatrix} v_0(t) \\ \vdots \\ v_M(t) \end{pmatrix} \quad (65)$$

where

$$p_m(t) = p(m \Delta x, t) \quad (66)$$

$$v_m(t) = v_x(m \Delta x, t) . \quad (67)$$

If the domain of x is of length ℓ and it is assumed subdivided into M subintervals then \underline{p} and \underline{v} are $M + 1$ dimensional and the system of

equations (62), (63) are replaced by the $2(M - 1)$ equations,

$$\frac{dp_m}{dt} = -c(m \Delta x) \frac{v_{m+1} - v_{m-1}}{2 \Delta x}, \quad m = 1, \dots, M - 1 \quad (68)$$

$$\frac{dv_m}{dt} = -c(m \Delta x) \frac{p_{m+1} - p_{m-1}}{2 \Delta x}, \quad m = 1, \dots, M - 1 \quad (69)$$

and four equations for p_o , p_m , v_o , and v_m . These last variables are usually separated from (68) and (69) since their corresponding equation forms are dependent upon the appropriate boundary conditions.

The boundary conditions appropriate to the system (68), (69) are obviously $2(M + 1)$ in number: the $2(M + 1)$ initial values if the system is completely homogeneous or $2(M - 1)$ initial conditions for the equation (68), (69) and $p_o(t)$, $p_m(t)$, $v_o(t)$, $v_m(t)$ if the system is assumed excited by its boundary values. The solution to this system is ideally carried out on an analog computer using separate interconnected integrators for each of the variables $p_m(t)$, $v_m(t)$; the number of integrators is large however, and the setup and scaling problems involved are very cumbersome to handle for all but very small values of M . A hybrid machine with an automatic inter-connection matrix, potsetting and error control would be theoretically the best solution but no large machine meeting these requirements presently exists or can be dedicated to so restricted a set of problems. Thus digital computers are usually employed to sequentially integrate such systems. The process is slow (relative to true problem time) even for the fastest, large-core machines now in existence. However, it is accurate and refinement of grid procedure can be made automatic.

Digitally, any of the common methods^[9] may be employed to integrate the ordinary differential equations (68), (69) but the accuracy is

clearly limited to the order of $(\Delta t + \Delta x)$. The use of a small time increment Δt is also required since stability requires that $\Delta t/(\Delta x)^2$ be bounded as $\Delta x, \Delta t$ tend to zero. The simplicity of the scheme together with the fact that only nearest-neighbor interactions are involved make the scheme quite useful if real-time computations are of secondary interest.

2. Explicit Scheme

An explicit scheme which is slightly more accurate as well as stable (provided $(c \Delta t / \Delta x) < 1$) is [10]

$$\frac{p_m^{n+1} - p_m^n}{\Delta t} = \frac{c}{\Delta x} (v_{m+\frac{1}{2}}^n - v_{m-\frac{1}{2}}^n) \quad (70)$$

$$\frac{v_{m-\frac{1}{2}}^{n+1} - v_{m-\frac{1}{2}}^n}{\Delta t} = \frac{c}{\Delta x} (p_m^{n+1} - p_{m-1}^{n+1}) \quad (71)$$

Here fractional subscripts (purely a notational device) have been introduced in order to avoid double space intervals $2 \Delta x$. The scheme (70, 71) can be arranged into purely explicit form as

$$p_m^{n+1} = p_m^n + \zeta (v_{m+\frac{1}{2}}^n - v_{m-\frac{1}{2}}^n) \quad (72)$$

$$\begin{aligned} v_{m-\frac{1}{2}}^{n+1} = v_{m-\frac{1}{2}}^n &+ \zeta (p_m^n - p_{m-1}^n) \\ &+ \zeta^2 (v_{m-\frac{1}{2}}^n - 2v_{m-1}^n + v_{m-3/2}^n) \end{aligned} \quad (73)$$

where

$$\zeta = \frac{c \Delta t}{\Delta x} \quad (74)$$

This scheme has been extensively used and can be generalized to three dimensions. The result is the system

$$\begin{aligned}
p(\ell, m, n, r+1) &= p(\ell, m, n, r) \\
&+ \zeta_x (v_x(\ell + \frac{1}{2}, m, n, r) - v_x(\ell - \frac{1}{2}, m, n, r)) \\
&+ \zeta_y (v_y(\ell + \frac{1}{2}, m, n, r) - v_y(\ell, m - \frac{1}{2}, n, r)) \\
&+ \zeta_z (v_z(\ell, m, n + \frac{1}{2}, r) - v_z(\ell, m, n - \frac{1}{2}, r)) \quad (75)
\end{aligned}$$

$$\begin{aligned}
v_x(\ell - \frac{1}{2}, m, n, r+1) &= v_x(\ell - \frac{1}{2}, m, n, r) \\
&+ \zeta_x (p(\ell, m, n, r) - p(\ell - 1, m, n, r)) \\
&+ \zeta_x^2 (v_x(\ell - \frac{1}{2}, m, n, r) - 2v_x(\ell - 1, m, n, r) \\
&+ v_x(\ell - 3/2, m, n, r)) \quad (76)
\end{aligned}$$

$$\begin{aligned}
v_z(\ell, m - \frac{1}{2}, n, r+1) &= v_y(\ell, m - \frac{1}{2}, n, r) \\
&+ \zeta_y (p(\ell, m, n, r) - p(\ell, m - 1, n, r)) \\
&+ \zeta_y^2 (v_y(\ell, m - \frac{1}{2}, n, r) - 2v_y(\ell, m - 1, n, r) \\
&+ v_z(\ell, m - 3/2, n, r)) \quad (77)
\end{aligned}$$

$$\begin{aligned}
v_z(\ell, m, n - \frac{1}{2}, r+1) &= v_z(\ell, m, n - \frac{1}{2}, r) \\
&+ \zeta_z (p(\ell, m, n, r) - p(\ell, m, n - 1, r)) \\
&+ \zeta_z^2 (v_z(\ell, m, n - \frac{1}{2}, r) - 2v_z(\ell, m, n - 1, r) \\
&+ v_z(\ell, m, n - 3/2, r)) \quad (78)
\end{aligned}$$

where

$$\begin{aligned}
p(\ell, m, n, r) &= p(\ell \Delta x, m \Delta y, n \Delta z, r \Delta t), \\
\zeta_x &= \frac{c \Delta t}{\Delta x}, \quad \zeta_y = \frac{c \Delta t}{\Delta y}, \quad \zeta_z = \frac{c \Delta t}{\Delta z}. \quad (79)
\end{aligned}$$

Computational experience on a CDC 6500 computer with this scheme has shown that: (1) The error is only slightly less than the scheme (67), (68); (2) The scheme is considerably more stable than (67), (68)

especially when the coefficients $\Delta x, \Delta y, \Delta z$ are not equal and/or when the coefficient $c(\underline{x})$ varies appreciably. This is because a slight amount of predictor-corrector behavior is present in that an improved updated value of the pressure is used on the right hand sides of (76,77,78).

(3) The computation time is much longer than the generalization to three dimensions of (68,69).

In view of the experience (3) an implicit scheme is very desirable.

3. Solution of the Diffusion Equation Implicit System

In Section B an implicit scheme for solution of the diffusion equation was presented; the equations can be solved very efficiently as follows: Writing (57) in the form

$$-A_m T_{m+1}^{n+1} + B_m T_m^{n+1} - C_m T_{m-1}^{n+1} = D_m, \quad m = 1, \dots, M-1 \quad (80)$$

it follows that

$$A_m = \frac{\sigma}{2(\Delta x)^2}, \quad B_m = \frac{\sigma}{(\Delta x)^2} + \frac{1}{\Delta t}, \quad C_m = A_m, \quad (81)$$

$$D_m = \left(\frac{1}{\Delta t} - \frac{\sigma}{(\Delta x)^2} \right) T_m^n + \frac{\sigma}{2(\Delta x)^2} T_{m+1}^n + \frac{\sigma}{2(\Delta x)^2} T_{m-1}^n.$$

In the above relations D_m is assumed completely known at all positions m at time state $n = 0$. In addition, the conditions at $x = 0$ and $x = M\Delta x$ are either assumed known at all times or they are assumed known at time $t = 0$ and subsequently they satisfy a linear-discrete relationship equivalent to the usual Sturm-Louville problem. For the present, assuming the simplest case with homogeneous conditions (46) and (47), the system of equations (80) together with the left hand boundary condition (46) determine a one-parameter family of solutions, i.e., two sets of quantities

E_m and F_m are desired which for any member T_m^n satisfy the relation

$$T_m^{n+1} = E_m T_{m+1}^{n+1} + F_m . \quad (82)$$

If (82) is true for any member it is true for $m = 0$ which requires by virtue of (47) that

$$E_0 = 0, \quad F_0 = 0 . \quad (83)$$

If now

$$T_{m-1}^{n+1} = E_{m-1} T_m^{n+1} + F_{m-1} \quad (84)$$

is substituted into (80) the result may be written as

$$T_m^{n+1} = \frac{A_m}{B_m - C_m E_{m-1}} T_{m+1}^n + \frac{D_m + C_m F_{m-1}}{B_m - C_m E_{m-1}} . \quad (85)$$

This in turn may be equated to (81) to obtain

$$E_m = \frac{A_m}{B_m - C_m E_{m-1}} , \quad m = 1, 2, \dots \quad (86)$$

$$F_m = \frac{A_m}{B_m - C_m E_{m-1}} , \quad m = 1, 2, \dots . \quad (87)$$

Starting with $m = 1$ and the values (83), the E_m and F_m can be calculated sequentially up to $m = M-1$. Once all the E_m, F_m are known (82) allows the calculation of T_m^{n+1} sequentially in decreasing m , ($m = M-1, M-2, \dots, 1$) since for $m = M$ the temperature T_{m+1}^{n+1} is known by (48).

This particular algorithm also has the advantage that the parameters A_m through F_m lie within a narrow dynamic range; it can be shown using the relations (81) that if $E_m \leq 1$ then

$$0 \leq E_m \leq \frac{A_m}{B_m - C_m} < \frac{A_m}{A_m} = 1 .$$

Since by (83) $E_0 = 0$, E_m lies between 0 and 1. Further, since the solution of the parabolic diffusion equation is analytic the solution T_m^n must be bounded and this with the bound on E_m requires that F_m also be reasonably bounded by (82).

The greatest advantage of this particular implicit algorithm solution is the saving in computation; it requires only three multiplications and two divisions per space step, per time step. It is a direct result of the matrix A in (61) being tri-diagonal in form. The corresponding solution by direct matrix inversion, which would only need be done once if the coefficients are independent of n , would require in addition to the inversion procedure - the multiplication of a matrix by a column vector - which would be M multiplications per space step, per time step.

4. Solution of the One-Dimensional Wave Equation Implicit System

The above algorithm for the diffusion equation can be extended to the wave equation in certain cases. For the one-dimensional wave equation (62), (63) a symmetric implicit scheme is

$$\frac{p_m^{n+1} - p_m^n}{\Delta t} = \frac{c}{2 \Delta x} (v_{m+\frac{1}{2}}^n - v_{m-\frac{1}{2}}^n + v_{m+\frac{1}{2}}^{n+1} - v_{m-\frac{1}{2}}^{n+1}) \quad (88)$$

$$\frac{v_{m-\frac{1}{2}}^{n+1} - v_{m-\frac{1}{2}}^n}{\Delta t} = \frac{c}{2 \Delta x} (p_m^{n+1} - p_{m-1}^{n+1} + p_{m-1}^n - p_{m-1}^n) \quad (89)$$

To investigate the stability of such schemes the Von Neumann theory has been generalized by Kreiss^[11] and others.^[11-15] Reference 4 summarizes the technique. Assuming that a suitable norm has been chosen for the domain of solutions of (88, 89) and that hence the Fourier analysis may

be applied to the system, let

$$\begin{bmatrix} p_m^n \\ v_m^n \end{bmatrix} = \sum_{k=-\infty}^{\infty} H_O(k) \begin{bmatrix} \hat{p}_k^n \\ \hat{v}_k^n \end{bmatrix} e^{jk(m \Delta x)} \quad (90)$$

$$\begin{bmatrix} p_m^{n+1} \\ v_m^{n+1} \end{bmatrix} = \sum_{k=-\infty}^{\infty} H_1(k) \begin{bmatrix} \hat{p}_k^{n+1} \\ \hat{v}_k^{n+1} \end{bmatrix} e^{jk(m \Delta x)} \quad (91)$$

If (90) is substituted into (88, 89) the result may be written as (after cancellation of the common factor $\exp(jk(m \Delta x))$ from all terms of the equations)

$$H_1 \begin{bmatrix} \hat{p}_k^{n+1} \\ \hat{v}_k^{n+1} \end{bmatrix} - H_O \begin{bmatrix} \hat{p}_k^n \\ \hat{v}_k^n \end{bmatrix} = 0 \quad (92)$$

where H_1 and H_O are the square matrices

$$H_1(k) = \frac{1}{\Delta t} \begin{bmatrix} 1 & , & -j \frac{c \Delta t}{\Delta x} \sin k \frac{\Delta x}{2} \\ -j \frac{c \Delta t}{\Delta x} e^{-jk \frac{\Delta x}{2}} \sin k \frac{\Delta x}{2} & , & e^{-jk \frac{\Delta x}{2}} \end{bmatrix} \quad (93)$$

$$H_O(k) = \frac{1}{\Delta t} \begin{bmatrix} 1 & , & j \frac{c \Delta t}{\Delta x} \sin k \frac{\Delta x}{2} \\ j \frac{c \Delta t}{\Delta x} e^{-jk \frac{\Delta x}{2}} \sin k \frac{\Delta x}{2} & , & e^{-jk \frac{\Delta x}{2}} \end{bmatrix} \quad (94)$$

The gain matrix $G(k, \Delta x, \Delta t)$ is then defined as

$$G(k, \Delta x, \Delta t) = (H_1)^{-1} H_O \quad (95)$$

and the general Von Neumann condition due to Richtmeyer^[4] may be stated as follows:

A sufficient condition for stability (and by the Lax equivalence theorem, for convergence) of a two-level difference scheme with transform matrices H_1 and H_0 is that the gain matrix

$$\left[G(k, \Delta x, \Delta t) \right]^n \quad \text{for} \quad \left\{ \begin{array}{l} 0 < \Delta t < \tau \\ 0 \leq n \Delta t \leq T \\ \text{all } k \text{ in } (90, 91) \end{array} \right\} \quad (96)$$

be uniformly bounded.

The matrix G for the present problem can be written as

$$G(k, \Delta x, \Delta t) = \frac{1}{1+a^2/4} \begin{pmatrix} 1-a^2/4 & ja \\ ja & 1-a^2/4 \end{pmatrix} \quad (97)$$

where

$$a = \frac{2c \Delta t}{\Delta x} \sin k \frac{\Delta x}{2} \quad (98)$$

It satisfies

$$G^*G - GG^* = 0 \quad (99)$$

and is therefore unitary.^[13] Hence G is always bounded and the scheme (88,89) is unconditionally stable. The system may be solved by the same scheme as previously shown for the diffusion equation since it is also tri-diagonal. In order to guarantee the non-singularity of the matrices involved it is easier to deal with integer subscripts; (88,89) can be written as

$$p_m^{n+1} - p_m^n = - \frac{c \Delta t}{4 \Delta x} (v_{m+1}^n - v_{m-1}^n + v_{m+1}^{n+1} - v_{m-1}^{n+1}) \quad (100)$$

$$v_m^{n+1} - v_m^n = - \frac{c \Delta t}{4 \Delta x} (p_{m+1}^n - p_{m-1}^n + p_{m+1}^{n+1} - p_{m-1}^{n+1}) . \quad (101)$$

The scalar equation (80) becomes the vector equation

$$-A_m \underline{u}_{m+1}^{n+1} + B_m \underline{u}_m^{n+1} - C_m \underline{u}_{m-1}^{n+1} = \underline{d}_m \quad (102)$$

where

$$\underline{u}_m^{n+1} = \begin{pmatrix} p_m^{n+1} \\ v_m^{n+1} \end{pmatrix} , \quad \underline{d}_m = \begin{pmatrix} p_m^n - \frac{c \Delta t}{4 \Delta x} (v_{m+1}^n - v_{m-1}^n) \\ v_m^n - \frac{c \Delta t}{4 \Delta x} (p_{m+1}^n - p_{m-1}^n) \end{pmatrix} \quad (103)$$

and

$$A = \begin{pmatrix} 0 & \frac{c \Delta t}{4 \Delta x} \\ \frac{c \Delta t}{4 \Delta x} & 0 \end{pmatrix} , \quad B = \begin{pmatrix} 1 & 0 \\ 0 & 1 \end{pmatrix} , \quad C = -A . \quad (104)$$

Again letting

$$\underline{u}_{m+1} = E_m \underline{u}_m + \underline{f}_m , \quad m = 0, 1, \dots, M \quad (105)$$

the results

$$E_m = (B_m - C_m E_{m-1})^{-1} A_m , \quad m = 1, \dots, M-1 \quad (106)$$

$$\underline{f}_m = (B_m - C_m E_{m-1})^{-1} (\underline{d}_m - C_m \underline{f}_{m-1}) , \quad m = 1, \dots, M-1 \quad (107)$$

are obtained. To solve the system suppose the matched boundary condition is applied at $x = M \Delta x$ in the form

$$\begin{pmatrix} p_{m-1} \\ v_{m-1} \end{pmatrix} = \begin{pmatrix} 0 & -1 \\ 1 & 0 \end{pmatrix} \begin{pmatrix} p_m \\ v_m \end{pmatrix} . \quad (108)$$

and a homogeneous velocity condition at $x = 0$ in the form

$$\begin{pmatrix} p_0 \\ v_0 \end{pmatrix} = \begin{pmatrix} p_0^{(n)} \\ 0 \end{pmatrix} , \quad p_0^{(n)} \text{ given.} \quad (109)$$

The "matched" condition (108) is clearly only an approximation to the correct condition for large Δx , but becomes progressively better as $\Delta x \rightarrow 0$.

This closed-trough problem has been programmed on the IBM 360 with the following results:

- A. The computation time for equivalent Δx is reduced by an order of magnitude over the scheme (73,74)
- B. The accuracy for a given Δx is considerably poorer than obtained by either (68,69) or (73,74) but this is thought to be because the boundary condition (108) is properly simulated only in the limit of $\Delta x \rightarrow 0$. Because of (A) above a much more rapid refinement of Δx is possible leading to very satisfactory results.
- C. Variable parameters are routinely handled and no instabilities were observed for reasonable parameter variations.

5. Three-Dimensional Implicit Formulation; Fractional Step Algorithm.

The results obtained for the one-dimensional implicit scheme for the wave equation, particularly the increase in computing speed, make a three-dimensional implicit scheme highly desirable. Therefore, a straightforward generalization of (100,101) was considered. The resulting scheme is

$$\begin{aligned}
 & p(\ell, m, n, r+1) - p(\ell, m, n, r) \\
 &= -\frac{\Delta t c(\ell, m, n)}{4} \left[\frac{1}{\Delta x} (v_x(\ell+1, m, n, r) - v_x(\ell-1, m, n, r) \right. \\
 &\quad \left. + v_x(\ell+1, m, n, r+1) - v_x(\ell-1, m, n, r+1) \right. \\
 &\quad + \frac{1}{\Delta y} (v_y(\ell, m+1, n, r) - v_y(\ell, m-1, n, r) + v_y(\ell, m+1, n, r+1) \\
 &\quad \left. - v_y(\ell, m-1, n, r+1)) \right. \\
 &\quad \left. + \frac{1}{\Delta z} (v_z(\ell, m, n+1, r) - v_z(\ell, m, n-1, r) + v_z(\ell, m, n+1, r+1) \right. \\
 &\quad \left. - v_z(\ell, m, n-1, r+1)) \right] \quad (110)
 \end{aligned}$$

$$\begin{aligned}
v_x(\ell, m, n, r+1) - v_x(\ell, m, n, r) = \\
- \frac{\Delta t c(\ell, m, n)}{4 \Delta x} (p(\ell+1, m, n, r) - p(\ell-1, m, n, r) + p(\ell+1, m, n, r+1) \\
- p(\ell-1, m, n, r+1)) \quad (111)
\end{aligned}$$

$$\begin{aligned}
v_y(\ell, m, n, r+1) - v_y(\ell, m, n, r) = \\
- \frac{\Delta t c(\ell, m, n)}{4 \Delta y} (p(\ell, m+1, n, r) - p(\ell, m-1, n, r) + p(\ell, m+1, n, r+1) \\
- p(\ell, m-1, n, r+1)) \quad (112)
\end{aligned}$$

$$\begin{aligned}
v_z(\ell, m, n, r+1) - v_z(\ell, m, n, r) = \\
- \frac{\Delta t c(\ell, m, n)}{4 \Delta z} (p(\ell, m, n+1, r) - p(\ell, m, n-1, r) + p(\ell, m, n+1, r+1) \\
- p(\ell, m, n-1, r+1)) \quad (113)
\end{aligned}$$

where as before $p(\ell, m, n, r) = p(\ell \Delta x, m \Delta y, n \Delta z, r \Delta t)$, etc. This algorithm is again unconditionally stable as $G(\underline{k}, \Delta x, \Delta y, \Delta z, \Delta t)$ is unitary but the tri-diagonal behavior has been lost; the matrix H_1 is of the form

$$H_1 = \begin{bmatrix} 1, j \frac{c \Delta t}{2 \Delta x} \sin k_1 \Delta x, j \frac{c \Delta t}{2 \Delta y} \sin k_2 \Delta y, j \frac{c \Delta t}{2 \Delta z} \sin k_3 \Delta z \\ j \frac{c \Delta t}{2 \Delta x} \sin k_1 \Delta x, 1, o, o \\ j \frac{c \Delta t}{2 \Delta y} \sin k_2 \Delta y, o, 1, o \\ j \frac{c \Delta t}{2 \Delta z} \sin k_3 \Delta z, o, o, 1 \end{bmatrix} \frac{1}{\Delta t} \quad (114)$$

which is penta-diagonal. Several problems were solved using this algorithm; it is highly accurate, stable under quite general conditions, but is relatively slow and requires a great deal of programming effort to use in the application being studied.

Many investigators have proposed schemes for solving the three-dimensional wave equation efficiently while maintaining stability. The

Soviet mathematicians Bagunovskii and Goduno,^[16] and later^[17] Yanenko have used partitions of the spacial operator for the various derivatives, applying each operator for fraction of the total time step Δt . Douglas and Gunn^[18] proposed a very general scheme for any number of space dimensions which involves p-tridiagonal calculations for a p-th order cartesian system. Douglas and Gunn's paper contains examples for the diffusion equation; the procedure has been applied to the system (9,10,11, 12) of Chapter II under the symmetric condition $\Delta x = \Delta y = \Delta z$. The resulting system of equations -- which is quite long -- is shown in Appendix A. The system can be shown to be unconditionally stable if the coefficient c is a constant; for variable c considerable computational experience seems to indicate considerable margin of stability exists if the variations in c do not exceed five percent.

6. The Fast Fourier Transform Technique for Constant Coefficients

In the special case that the velocity c may be treated as a constant over the domain of integration, the use of the digital discrete Fast Fourier Transform (FFT) algorithm^[19] provides a simple, elegant solution of either explicit or implicit algorithm schemes. The application of the discrete Fourier transform to this problem in the manner to follow is new.

Since implicit algorithms in general, as shown above, exhibit superior stability properties and when properly formulated greatly reduced computation requirements, the FFT algorithm was applied to their solution.

The implicit scheme (100,101) will be solved to illustrate the method. Assume that the initial states

$$\underline{u}_m^o = \begin{pmatrix} p_m^o \\ v_m^o \end{pmatrix}, \quad m = 0, 1, \dots, M-1 \quad (115)$$

are known and that the number of sample points M , including boundary points is an integral multiple of two. This restriction is not absolutely necessary but its imposition guarantees the maximum computational efficiency for the FFT algorithm. The discrete set of values may then be represented by the finite inverse Fourier transform as

$$\begin{pmatrix} p_m^o \\ v_m^o \end{pmatrix} = \sum_{k=0}^{M-1} \begin{pmatrix} \hat{p}_k^o \\ \hat{v}_k^o \end{pmatrix} e^{j \frac{2\pi k}{M} m} \quad (116)$$

where the spectral functions \hat{p}_k^o , \hat{v}_k^o are obtained by the process

$$\begin{pmatrix} \hat{p}_k^o \\ \hat{v}_k^o \end{pmatrix} = \frac{1}{M} \sum_{m=0}^{M-1} \begin{pmatrix} p_m^o \\ v_m^o \end{pmatrix} e^{-j \frac{2\pi m}{M} k} \quad (117)$$

This result, coupled with the fact that, by (116)

$$\begin{pmatrix} p_{m \pm \beta}^o \\ v_{m \pm \beta}^o \end{pmatrix} = \sum_{k=0}^{M-1} \begin{pmatrix} \hat{p}_k^o \\ \hat{v}_k^o \end{pmatrix} e^{j \frac{2\pi k}{M} (m \pm \beta)}, \quad \beta = \text{integer}, \quad (118)$$

leads to the following equation if the transform is applied to (100,101)

under the assumption that c is a constant.

$$\frac{1}{\Delta t} \sum_{k=0}^{M-1} \left[\begin{pmatrix} 1 & -j \frac{c \Delta t}{2 \Delta x} \sin \frac{2\pi k}{M} \\ -j \frac{c \Delta t}{2 \Delta x} \sin \frac{2\pi k}{M} & 1 \end{pmatrix} \begin{pmatrix} \hat{p}_k^1 \\ \hat{v}_k^1 \end{pmatrix} - \begin{pmatrix} 1 & j \frac{c \Delta t}{2 \Delta x} \sin \frac{2\pi k}{M} \\ j \frac{c \Delta t}{2 \Delta x} \sin \frac{2\pi k}{M} & 1 \end{pmatrix} \begin{pmatrix} \hat{p}_k^o \\ \hat{v}_k^o \end{pmatrix} \right] e^{j \frac{2\pi k}{M} m} = 0 \quad (119)$$

This can be true for arbitrary m only if the large bracket term vanishes.

Hence defining the matrices

$$G_1(k, \Delta x, \Delta t) = \begin{pmatrix} 1 & -\alpha \\ -\alpha & 1 \end{pmatrix}, \quad (120)$$

and

$$G_0(k, \Delta x, \Delta t) = \begin{pmatrix} 1 & \alpha \\ \alpha & 1 \end{pmatrix}, \quad (121)$$

where

$$\alpha = j \frac{c \Delta t}{2 \Delta x} \sin \frac{2\pi k}{M}, \quad (122)$$

it follows that

$$\begin{aligned} \begin{pmatrix} \hat{p}_k^1 \\ \hat{v}_k^1 \end{pmatrix} &= (G_1)^{-1} G_0 \begin{pmatrix} \hat{p}_k^0 \\ \hat{v}_k^0 \end{pmatrix} \\ &= G_{10} \begin{pmatrix} \hat{p}_k^0 \\ \hat{v}_k^0 \end{pmatrix} \end{aligned} \quad (123)$$

or

$$\begin{pmatrix} \hat{p}_m^1 \\ \hat{v}_m^1 \end{pmatrix} = \sum_{k=0}^{M-1} G_{10}(k, \Delta x, \Delta t) \begin{pmatrix} \hat{p}_k^0 \\ \hat{v}_k^0 \end{pmatrix} e^{j \frac{2\pi k}{M} m}. \quad (124)$$

Also, by induction

$$\begin{pmatrix} \hat{p}_m^n \\ \hat{v}_m^n \end{pmatrix} = \sum_{k=0}^{M-1} (G_{10}(k, \Delta x, \Delta t))^n \begin{pmatrix} \hat{p}_k^0 \\ \hat{v}_k^0 \end{pmatrix} e^{j \frac{2\pi k}{M} m}. \quad (125)$$

Hence apart from the calculation of the powers of $G_{10}(k)$ the entire process involves only the application of the FFT algorithm at the time states $n=0$ and the desired state n . The powers of $G_{10}(k)$ may be most efficiently evaluated for large n by use of Sylvester's theorem. [20]

The procedure is readily extendable to three dimensions; the penta-diagonal scheme (110,111,112,113) has the solution

$$\underline{u}(\ell, m, n, r) = \begin{pmatrix} p(\ell, m, n, r) \\ v_x(\ell, m, n, r) \\ v_y(\ell, m, n, r) \\ v_z(\ell, m, n, r) \end{pmatrix}$$

$$= \sum_{k_3=0}^{N-1} \sum_{k_2=0}^{M-1} \sum_{k_1=0}^{L-1} (G_{10}(k_1, k_2, k_3))^r \begin{bmatrix} \hat{p}(k_1, k_2, k_3, 0) \\ \hat{v}_x(k_1, k_2, k_3, 0) \\ \hat{v}_y(k_1, k_2, k_3, 0) \\ \hat{v}_z(k_1, k_2, k_3, 0) \end{bmatrix}$$

$$\cdot e^{j2\pi \left(\frac{k_1 \ell}{L} + \frac{k_2 m}{M} + \frac{k_3 n}{N} \right)}, \quad (126)$$

where

$$G_{10}(k_1, k_2, k_3) = G_1(k_1, k_2, k_3)^{-1} G_0(k_1, k_2, k_3)$$

$$= \begin{pmatrix} 1 & \alpha & \beta & \gamma \\ \alpha & 1 & 0 & 0 \\ \beta & 0 & 1 & 0 \\ \gamma & 0 & 0 & 1 \end{pmatrix}^{-1} \begin{pmatrix} 1 & -\alpha & -\beta & -\gamma \\ -\alpha & 1 & 0 & 0 \\ -\beta & 0 & 1 & 0 \\ -\gamma & 0 & 0 & 1 \end{pmatrix}, \quad (127)$$

and

$$\alpha = \frac{jc \Delta t}{2 \Delta x} \sin \frac{2\pi k_1}{L}, \quad \beta = \frac{jc \Delta t}{2 \Delta y} \sin \frac{2\pi k_2}{M}, \quad \gamma = \frac{jc \Delta t}{2 \Delta z} \sin \frac{2\pi k_3}{N}. \quad (128)$$

There is, of course, an unfortunate oversight in the above manipulations: the equations (100,101) involve the boundary conditions at $m=0$ and $m=M$ which are outside of the indexing of (100,101). The same must be true in the above formulation so that what has been solved is the system in which $p(l,m,n,o), v(l,m,n,o)$ are known and the tacit homogeneous boundary conditions $p(-1,m,n,r) = p(L,m,n,r) = p(l,-1,n,r) = p(l,M,n,r) = 0$, etc., have been assumed. (A homogeneous nearest-neighbor shell of boundary conditions on p,v enclosing the parallelepiped $L\Delta x$ by $M\Delta y$ by $N\Delta z$.) The method of solution can be extended to handle this case by making use of the theorem by Courant and Hilbert,^[31] wherein it is shown that: homogeneous differential equations with non-homogeneous boundary conditions are essentially equivalent to non-homogeneous differential equations with homogeneous boundary conditions. Usually the application of the theorem requires in addition that certain smoothness conditions be satisfied by the boundary conditions, such as differentiability, etc. The theorem is a natural extension to partial-differential equations of the familiar result that the solution to an inhomogeneous differential equation consists of a solution to the homogeneous equation which satisfies the same initial conditions plus any particular integral of the inhomogeneous equation satisfying homogeneous initial conditions. The theorem is restricted to linear equations.

To apply this theorem to the problem under discussion several methods are possible: (1) the application of the Lagrangian multipliers

method, (2) the eigenfunction expansion method in which the projection of the boundary conditions on the complete set of eigenfunctions appropriate to the linear homogeneous operator equivalent to (100,101) is used to augment the operator, (3) the use of symbolic functions to include the boundary conditions.

Method (3) is the easiest method to apply in the present case. The preliminary details of the method proceed by finding the adjoint operator A^* for the system assuming homogeneous boundary conditions; the scalar product

$$\langle A^* \underline{u}, A_e \rangle = \langle A^*, A_e \underline{u} \rangle \quad (129)$$

is formed where A_e is an extended linear operator and the domain of \underline{u} now includes the non-homogeneous boundary values. The forced equality in (129) generally requires that A_e involve singularity functions; in the continuous differential equation case usually the Dirac function and its derivatives are involved. In the case of discrete systems the Kronecker function

$$\begin{aligned} \delta_{k,\ell} &= 1 & , & \ell = k \\ &= 0 & , & \ell \neq k \end{aligned} \quad (130)$$

replaces the Dirac function.

For the problem under consideration here let the solution at time state $n=0$ be denoted as \underline{u}^0 ,

$$\begin{aligned} \underline{u}^0 &= \underline{u}^0(m \Delta x) \\ &= \begin{pmatrix} p^0(m \Delta x) \\ v^0(m \Delta x) \end{pmatrix}. \end{aligned} \quad (131)$$

\underline{u}^0 is assumed to consist of two parts: a homogeneous part \underline{u}_h^0 which is

the solution to the linear difference equations (100,101) with

$$\begin{aligned}\underline{u}_h^0(m\Delta x) &= \underline{u}^0(m) \quad , \quad m \neq 0 \text{ or } M-1 \\ &= 0 \quad , \quad m=0 \text{ or } M-1, \end{aligned} \quad (132)$$

and an inhomogeneous part \underline{u}_i^0 which satisfies

$$\begin{aligned}\underline{u}_i^0(m\Delta x) &= 0 \quad , \quad m \neq 0 \text{ or } M-1, \\ &= \underline{u}_i^0(0) \quad , \quad m = 0, \\ &= \underline{u}_i^0(M-1) \quad , \quad m = M-1. \end{aligned} \quad (133)$$

The values $\underline{u}_i^0(0)$, $\underline{u}_i^0(M-1)$ are the boundary values for the problem and are assumed known a priori. In terms of equations (100,101) the predicted value \underline{u}_i^1 has component p_m^1 , v_m^1 satisfying

$$p_m^1 - p_m^0 = \frac{c\Delta t}{4\Delta x} (v_{m+1}^1 - v_{m-1}^1 + v_{m+1}^0 - v_{m-1}^0) + \delta_{m,0} p_0^0 + \delta_{m,M-1} p_{M-1}^0 \quad (134)$$

$$v_m^1 - v_m^0 = \frac{c\Delta t}{4\Delta x} (p_{m+1}^1 - p_{m-1}^1 + p_{m+1}^0 - p_{m-1}^0) + \delta_{m,0} v_0^0 + \delta_{m,M-1} v_{M-1}^0. \quad (135)$$

For the applications which require a linear combination boundary condition such as for example

$$p_0^0 + b v_0^0 = g^0, \quad (136)$$

where b is a constant - the matched boundary condition is an example - g^0 replaces p_0^0 in (134).

Now denoting the FFT of \underline{u}_i^0 as $\hat{\underline{u}}_i^0$ and of \underline{u}_i^1 as $\hat{\underline{u}}_i^1$ and applying the transform to (134,135) results in the matrix equation.

$$G_1 \hat{\underline{u}}_i^1 = G_0 \hat{\underline{u}}_i^0 + \underline{u}_i^0(0) + e^{\frac{-j 2\pi (M-1)}{M} k} \underline{u}_i^0(M-1). \quad (137)$$

By hypothesis \underline{u}_1^0 is zero for all m in (134,135) as far as the terms involved in $G_o \underline{u}_i^0$ are concerned so that

$$G_o \underline{u}_i^0 = 0 \quad (138)$$

and hence

$$\underline{u}_i^1 = G_1^{-1} \left\{ \underline{u}_i^0(0) + e^{\frac{-j2\pi(M-1)}{M} k} \underline{u}_i^0(M-1) \right\} \quad (139)$$

is the solution for the FFT spectrum of the inhomogeneous solution \underline{u}_i^1 at time state $n=1 \Delta t$. The solution for \underline{u}_h^1 is that already found in (122)

$$\begin{aligned} \underline{u}_h^1 &= G_1^{-1} G_o \underline{u}_h^0 \\ &= G_{10} \underline{u}_h^0 \end{aligned} \quad (140)$$

Since the FFT is a linear mapping operation the spectrum of the sum of two functions is the sum of the individual function spectra and hence

$$\begin{aligned} \underline{u}^1 &= \underline{u}_h^1 + \underline{u}_i^1 \\ &= G_{10} \underline{u}_h^0 + G_1^{-1} \underline{u}_i^0 \end{aligned} \quad (141)$$

The solution (141) represents the spectrum at time state $n=1$. To proceed to time state $n=2$ the process is again repeated except the homogeneous spectrum at time state $n=1$ is now taken to be the total solution spectrum given in (141). The equation (141) for time state $n=2$ thus becomes

$$\begin{aligned} \underline{u}^2 &= G_{10} \underline{u}^1 + G_1^{-1} \underline{u}_i^1 \\ &= G_{10} (G_{10} \underline{u}_h^0 + G_1^{-1} \underline{u}_i^0) + G_1^{-1} \underline{u}_i^1 \\ &= (G_{10})^2 \underline{u}_h^0 + G_{10} G_1^{-1} \underline{u}_i^0 + G_1^{-1} \underline{u}_i^1 \end{aligned} \quad (142)$$

where

$$\underline{\hat{u}}_i^1 = G_1^{-1} \left\{ \underline{u}_i^1(0) + e^{-j \frac{2\pi(M-1)}{M} k} \underline{u}_i^1(M-1) \right\} \quad (143)$$

By induction the general solution at time state $m\Delta t$ is given by

$$\underline{\hat{u}}^r = (G_{10})^n \underline{\hat{u}}_h^0 + (G_{10})^{n-1} G_1^{-1} \underline{\hat{u}}_i^0 + (G_{10})^{n-2} G_1^{-1} \underline{\hat{u}}_i^1 + \dots G_1^{-1} \underline{\hat{u}}_i^{n-1}. \quad (144)$$

An obvious thought is that since the matrix G_{10} is unitary and bounded and hence the homogeneous solution is stable independent of Δt , why not set $\Delta t = t$ the desired final time and iterate the process only once? The answer to this conjecture is negative. The system of equations (134,135) has a stability criterion which no longer is unconditional, depending in a complex way on the functional behavior of the boundary values. A moments thought leads to the realization that this must be so since only the homogeneous solution is effectively implicit; the inhomogeneous solution by (139) is explicit. Hence, the overall scheme is only semi-implicit and the magnitude of the Δt allowed depends on the magnitude of the increments in the boundary conditions between time steps.

Since the rate of change of the boundary values with time is linearly related to the frequency contained in the frequency spectrum of the boundary values, and as will be shown, the magnitude of the frequency must be low to guarantee proper non-dispersive simulation relative to the Shannon rate, the increments in the boundary conditions are generally quite small. This, coupled with the fact that only the inhomogeneous portion of the solution is not unconditionally stable, leads to the qualitative speculation that significant magnitude Δt may be used.

Computer solutions of the above algorithm have verified that such is indeed the case; quite stable and accurate results were obtained for Δt 's more than twice the size of that allowed by the scheme (71,72) and with reductions factors in computation time of greater than ten. Since the number of calculations for the three-dimensional problem is of the order of the cube of that for the one-dimensional problem, the use of schemes like (110,111,112,113) is only justified for problems in which the variation in the coefficients is so rapid that prohibitively small domains would be necessary in order to use the FFT scheme.

An additional advantage of the FFT scheme is that for problems of the type of primary interest in this work in which spacial partitioning is possible, the objective of integrating the wave equation is primarily to find the steady state generated values of the unknown boundaries of a parallelepiped in terms of known values over the other boundaries of the parallelepiped. (For example if the values were known on six surfaces of a parallelepiped, what are the steady state boundary values on the seventh surface?) By virtue of (126), this would correspond to inverting the transform only for a fixed two-dimensional plane so that only L M -point inverse transforms need be calculated rather than $(L \times N)$ M -point inverse transforms - a considerable saving of computational effort.

D. BAND-LIMITED PROPERTIES OF DISCRETE MODELS; SPACIAL SAMPLING DIMENSIONS

The above algorithms, whether continuous-discrete, explicit-discrete, or implicit-discrete, all require great amounts of numerical computation, the amount depending fundamentally on the spacial partition parameter

magnitudes Δx , Δy , Δz . In section A it was claimed that the element partition size Δx , in the one-dimensional case, was determined by the operating frequency whose simulation is to be studied and the sound speed c .

1. Cutoff Properties

The simple discretized wave equation (68,69) is an exact dual of the one-dimensional periodic, (in the constant coefficient case) mass-loaded line originally studied by Baden-Powell,^[5] Lord Kelvin^[23] and others. The propagation properties of such lines can be found by application of Floquet's theorem;^[24] if the assumed solutions

$$p(m \Delta x, t) = p_o e^{j\omega t - \gamma m \Delta x} \quad (145)$$

$$v(m \Delta x, t) = v_o e^{j\omega t - \gamma m \Delta x} \quad (146)$$

are substituted into (68,69) the determinantal equation for the propagation constant γ results,

$$(\sinh k \gamma \Delta x)^2 = -\frac{\omega^2}{c^2} (\Delta x)^2, \quad (147)$$

The form of the result (147) varies slightly depending on the finite difference formulation used but all discrete schemes result in the same qualitative behavior. For lossless propagation γ must be pure imaginary

$$\gamma = j\beta, \quad (148)$$

which gives in (147)

$$\begin{aligned} \sin \beta \Delta x &= \pm \sqrt{\left(\frac{\omega}{c} \Delta x\right)^2} \\ &= \pm \frac{\omega \Delta x}{c}. \end{aligned} \quad (149)$$

Real β solutions to (149) occur only if

$$\frac{\omega \Delta x}{c} \leq 1 \quad (150)$$

from which the cutoff frequency f_c is obtained as

$$f_c = \frac{c}{2\pi \Delta x} \quad (151)$$

Since the velocity c is an inherent property of the medium, if propagation of a monochromatic signal of frequency f is to be simulated, f should satisfy the inequality

$$f < \frac{c}{2\pi \Delta x} \quad (152)$$

Assuming a nominal velocity in sea water of 1500 meters/sec. this restricts Δx to be less than

$$\Delta x < \frac{239}{f} \quad (153)$$

For a frequency of 500 Hz., $\Delta x < .478$ meters, a very short distance.

The Shannon spacial sampling distance would be 1.5 meters so that the cutoff frequency restriction is more severe than the sampling restriction necessary to properly simulate the partial-differential equation.

2. Dispersion Properties

Equation (149) also indicates that the propagation is dispersive, i.e., the propagation constant β ,

$$\beta = \frac{1}{\Delta x} \sin^{-1} \left(\frac{\omega \Delta x}{c} \right) \quad (154)$$

is not a linear function of frequency except for very low frequencies. In order to make the argument of the arcsine in (154) small, Δx must be restricted to

$$\frac{\omega \Delta x}{c} \sim .1 \quad (155)$$

or

$$f \sim .1f_c \quad (156)$$

for non-dispersive simulation. Thus the restrictions on spacial increments are very severe, especially for very higher frequencies.

IV. COMPUTER RESULTS FOR THE MATCHED BOUNDARY CONDITION

A. PARTITIONING OF THE SPACIAL DOMAIN

The integration of the acoustic wave equation over large spacial domains in which the sound speed c varies with position is usually impractical to carry out on digital computers because of the huge amounts of core required for even crude finite-cell discretizations. To implement for example the algorithm (110,111,112,113) for a rectangular parallelepiped region consisting of $(25 \times 25 \times 10)$ spacial integration volumes requires a minimum of 50,000 words of memory just to store the required values of the variables. The coding of such a problem requires great amounts of indexing calculations which typically would require at least an additional 10,000 words of memory for the stored instructions, and temporary work storage. These requirements, especially in view of the fact that the parallelepiped would simulate a quite small physical volume for all but very low frequencies, has made ray theory and normal-mode model approximations the only practical methods being presently employed for investigating undersea acoustic propagation.

A problem of considerable interest is the calculation of the steady-state acoustic loss over great distance for various frequency acoustic waves in seawater wherein the sound speed c varies with position. As pointed out in the introduction, at present this calculation is carried out using ray theory to establish an approximate sound trajectory and then the loss is determined by integrating an empirical loss factor per unit length

along the trajectory.

In view of the practical limitations on core total storage size for available digital computers, the more fundamental direct integration of the wave equation is practical only if problems involving large spacial domains can be partitioned into relatively small spacial domains which can occupy core storage sequentially. The macro domain of $25 \times 25 \times 10$ micro domains, (micro domains will refer to fundamental small elements of size Δx by Δy by Δz) might constitute such a relatively small spacial domain.

B. THE INHOMOGENEOUS LINE IMPEDANCE

In one-dimensional uniform transmission line theory a finite section of line of length ℓ can simulate a portion of an infinite line provided the voltage and current at the ends of the line are constrained by the equations (see Fig. 3)

$$v(0,t) = Z_0 i_0(0,t) \quad (157)$$

$$v(\ell,t) = Z_0 i(\ell,t) \quad (158)$$

where Z_0 is called the characteristic impedance of the transmission line and is related to the series impedance per unit length Z_s and shunt admittance per unit length Y_{sh} by the relation

$$Z_0 = \sqrt{\frac{Z_s}{Y_{sh}}} \quad (158)$$

If the line is lossless and possesses only series L and shunt C elements the equations of the line are

$$C \frac{\partial v}{\partial t} = \frac{\partial i}{\partial x} \quad (159)$$

$$L \frac{\partial i}{\partial t} = - \frac{\partial V}{\partial t} \quad (160)$$

which are the complete duals of the equations (62) and (63). If steady state solutions are obtained for (159, 160) by assuming

$$\begin{pmatrix} v(x, t) \\ i(x, t) \end{pmatrix} = \text{Re} \left\{ \begin{pmatrix} \hat{V}(x) \\ \hat{I}(x) \end{pmatrix} e^{j\omega t} \right\} \quad (161)$$

the voltage and current phasors $\hat{V}(x)$, $\hat{I}(x)$ satisfy the relation

$$\frac{\hat{V}(x)}{\hat{I}(x)} = Z(x) \quad (162)$$

where $Z(x)$ is the complex impedance

$$Z(x) = Z_0 \frac{Z(\ell) \cos(\omega\sqrt{LC}(\ell-x)) + jZ_0 \sin(\omega\sqrt{LC}(\ell-x))}{Z_0 \cos(\omega\sqrt{LC}(\ell-x)) + jZ(\ell) \sin(\omega\sqrt{LC}(\ell-x))} \quad (163)$$

and

$$\begin{aligned} Z_0 &= \sqrt{\frac{Z_s}{Y_{sh}}} \\ &= \sqrt{\frac{L}{C}} \end{aligned} \quad (164)$$

Thus the characteristic impedance is a pure real constant only if the line is lossless or in the happy circumstance where

$$\frac{R}{L_s} = \frac{G}{C_{sh}}, \quad (165)$$

in which case though Z_s and Y_{sh} are both complex their phase angles are equal and thus cancel. In (163), $Z(\ell)$ is the terminating impedance of the line; if $Z(\ell)$ has the value Z_0 (163) gives the result

$$Z(x) = Z_0, \quad (166)$$

a pure constant. The general steady state solution to (159) may be written as

$$\hat{V}(x) = V^+ e^{-x\sqrt{Z_s Y_{sh}}} + V^- e^{x\sqrt{Z_s Y_{sh}}} ; \quad (167)$$

if the line is lossless this results in

$$\hat{V}(x) = V^+ e^{-j\omega\sqrt{LC}(\ell-x)} + V^- e^{j\omega\sqrt{LC}(\ell-x)} \quad (168)$$

$$\hat{I}(x) = \frac{\hat{V}(x)}{Z(x)} \quad (169)$$

where V^+ , V^- are the amplitudes of positive and negatively traveling waves respectively. Causally if a monochromatic disturbance is propagated in the positive x direction with amplitude V^+ the coefficient V^- will be zero unless there is a change in the characteristic parameters of the transmission line. In such a case reflection coefficient R can be defined by

$$\begin{aligned} R(x) &= \frac{V^- e^{j\omega\sqrt{LC}(\ell-x)}}{V^+ e^{j\omega\sqrt{LC}(\ell-x)}} \\ &= R(\ell) e^{-j2\omega\sqrt{LC}(\ell-x)} , \end{aligned} \quad (170)$$

and is a measure of the "non-match" of the line. If the parameters L and C are not constants but functions of position Shelkunoff^[25] has generalized the concept of reflection coefficient for inhomogeneous transmission lines in terms of the propagation constant $\gamma(x)$ where,

$$\begin{aligned} \gamma(x) &= \sqrt{Z_s(x) Y_{sh}(x)} \\ &= \alpha(x) + j\beta(x) . \end{aligned} \quad (171)$$

The derivation to follow deviates slightly from Shelkunoff's approach being specifically oriented to the acoustic problem at hand. Consider the two semi-infinite half spaces separated by the plane $x=0$. The region $x < 0$ is assumed to be homogeneous with constant parameters c , ρ , κ ,

the region $x > 0$ has slowly varying parameters $\kappa(x)$ and $c(x)$. Assuming steady state dependence of the form

$$\begin{pmatrix} p(x,t) \\ v(x,t) \end{pmatrix} = \text{Re} \left[\begin{pmatrix} \hat{p}(x) \\ \hat{v}(x) \end{pmatrix} e^{j\omega t} \right] \quad (172)$$

in equations (62,63) reduces the partial differential equations to the ordinary differential equations

$$\frac{d\hat{p}}{dx} = -j\omega\rho\hat{v} \quad (173)$$

$$\frac{d\hat{v}}{dx} = -j\omega\frac{\hat{p}}{\kappa} \quad (174)$$

For $x < 0$ the solution to this system for $P(x)$ assuming unit incident amplitude is

$$\hat{p}(x) = e^{-\gamma x} + R(x) e^{\gamma x} \quad (175)$$

where $R(x)$ is the reflection coefficient. Since the reflection coefficient at any point x_0 is a measure of the mismatch as it affects the behavior for $x < x_0$ of inhomogenities occurring for $x > x_0$, (175) as pointed out by Schelkunoff, provides a plane wave reflection coefficient which holds for any x . The propagation constant γ is obtained from (173) and (174) as

$$\begin{aligned} \gamma &= j\beta \\ &= j\frac{\omega}{c(x)} \end{aligned} \quad (176)$$

Defining the general impedance at a point as

$$Z(x) = \frac{\hat{p}(x)}{\hat{v}(x)} \quad (177)$$

and using (173), the result may be written as

$$\begin{aligned}
 Z(x) &= \frac{\hat{P}}{-\frac{1}{j\omega\rho} \left(\frac{d\hat{P}}{dx} \right)} \\
 &= \frac{-j\omega\rho}{\frac{d}{dx} (\log \hat{P})} \quad .
 \end{aligned} \tag{178}$$

From (175)

$$\begin{aligned}
 Z(x) &= -j\omega\rho \frac{\hat{P}}{\left(\frac{d\hat{P}}{dx} \right)} \\
 &= \frac{-j\omega\rho}{-\gamma (e^{-\gamma x} - R(x)e^{\gamma x}) + R'(x) e^{\gamma x}} \\
 &\simeq \frac{j\omega\rho}{\gamma} \left(\frac{e^{-\gamma x} + R(x) e^{\gamma x}}{e^{-\gamma x} - R(x) e^{\gamma x}} \right)
 \end{aligned} \tag{179}$$

if $R(x) \ll 1$ and slowly varying. By virtue of the continuity of the ratio \hat{P}/V the function (179) is continuous across an interface boundary even if $\kappa(x)$ and $c(x)$ are discontinuous. By virtue of the fact that

$$\frac{d\hat{P}}{dx} = \hat{P} \frac{d}{dx} (\log \hat{P}), \tag{180}$$

the equation

$$\frac{d^2 \hat{P}}{dx^2} - \gamma^2 \hat{P} = 0 \tag{181}$$

may be written as

$$\begin{aligned}
 0 &= \frac{d}{dx} \left(\frac{d\hat{P}}{dx} \right) - \gamma^2 \hat{P} \\
 &= \frac{d}{dx} \left(\hat{P} \frac{d}{dx} (\log \hat{P}) \right) - \gamma^2 \hat{P} \\
 &= \frac{d\hat{P}}{dx} \frac{d}{dx} (\log \hat{P}) + \hat{P} \frac{d^2}{dx^2} (\log \hat{P}) - \gamma^2 \hat{P}
 \end{aligned}$$

$$= \hat{P} \left(\frac{d}{dx} (\log \hat{P}) \right)^2 + \hat{P} \frac{d^2}{dx^2} (\log \hat{P}) - \gamma^2 \hat{P} \quad (182)$$

or

$$\frac{d^2}{dx^2} (\log \hat{P}) = \gamma^2 - \left(\frac{d}{dx} (\log \hat{P}) \right)^2. \quad (183)$$

By use of (178), equation (183) becomes

$$\frac{d}{dx} \left(\frac{-j\omega\rho}{Z(x)} \right) = \gamma^2 \left(\frac{-j\omega\rho}{Z(x)} \right), \quad (184)$$

or

$$\frac{dZ}{dx} + \frac{\gamma^2 Z^2}{j\omega\rho} + j\omega\rho = 0. \quad (185)$$

This is a first-order non-linear equation for the impedance as a function of the propagation constant $\gamma(x)$. Alternately, (179) may be inverted to $R(x)$ in terms of $Z(x)$ and the result substituted in (183) to obtain the differential equation for the reflection coefficient

$$\frac{dR}{dx} - \frac{1}{2}(1-R^2) \frac{d}{dx} (\log (-j\gamma)) - 2\gamma R = 0, \quad (186)$$

a first-order non-linear equation of the Riccati type. If

$$\frac{1}{\gamma} \frac{d}{dx} (\log (-j\gamma)) \ll 1 \quad (187)$$

then (186) becomes

$$\frac{dR}{dx} \simeq -2\gamma R \quad (188)$$

which has the solution

$$R(x) = R(0) e^{-2 \int_0^x \gamma(\xi) d\xi}. \quad (189)$$

By virtue of (176) it can be seen that the effect in this case is simply a phase shift due to the length of path and to first order if $c(x)$ is smoothly

varying, (in a multi-dimensional problem) to cause a bending in the direction of the wave. [1]

C. THE MATCHED BOUNDARY-CONDITION DOMAIN

The matched termination or characteristic impedance has been extensively discussed in the excellent book by Brillouin [5]; he considers the general periodic structure lattices consisting of polyatomic molecules and derives an expression giving the characteristic impedance as a function of the periodic cell internal energy. He concludes his discussion by emphasizing the difficulty involved in theoretically defining the characteristic impedance for structures in which more than one dimension is involved or where more than nearest neighbor interactions are considered.

Here the objective was to partition variable-parameter, large domains in three-dimensions with the objective of computing steady state trajectories and pressure distributions. Consider Fig. 4 which shows a large domain consisting of 27 macro-domains. Each of the individual macro-domains consists of micro-domains; ($L=M=5$, $N=3$). As the simplest and most straight-forward method to apply, an Adams-Moulton predictor-corrector, with dynamic error control program called RKAM (see program listings), was used to integrate the explicit three-dimensional finite difference scheme equivalent to the one-dimensional scheme shown in (68,69). The procedure for integrating the large domain was to integrate the problem two ways and then to compare the results:

- (A) Integrate all $(3+1)(5)(5)(3)(27) = 8100$ equations in an overlay program until total steady state energy was attained for the entire large domain with characteristic boundary conditions

being employed on the outer-most boundaries. The energy source was placed asymmetrically in the interior of the innermost macro-domain in order to preclude any possible cancellation effects due to symmetry.

- (B) Integrate the 300 equations for the innermost macro-domain with the energy source located in the same position as (1) above. Characteristic boundary conditions were again employed except their location was the boundary of the innermost macro-domain. Upon reaching steady state conditions the FFT was employed to find equivalent oscillators for the pressure and three velocity components at each micro-domain on the surface of the inner-most macro-domain. These oscillators are then assumed to be the sources for the nearest neighbor adjacent micro-domains in the adjoining macro-domain, (the boundary micro-domains are considered to belong to both adjoining macro-domains.) In general, only the nearest neighbor macro-domains are considered in establishing steady state values for the boundaries. Since the medium is linear, the rms magnitude of the states on the boundaries can be found as the linear superposition of the quadratic contributions to the rms value of each adjacent macro-domain boundary effect. Proceeding macro-domain by macro-domain, the equivalent rms boundary values for the large domain were obtained.

The above procedures were performed on the Fleet Numerical Weather Facility CDC 6500 computer which carries out floating-point calculations to 14 decimal digits. In the case of constant coefficient domains the above two procedures produced results in excellent agreement considering the prodigious amounts of computation involved; the two computations were in agreement at all but a very small set of values to three decimal digits, There were no major discrepancies.

In the variable-parameter domain case the concept of matched characteristic impedance loses significance unless more than nearest neighbor interactions are considered because the characteristic impedance can be fundamentally defined only for periodic domains. However, when the variations in the velocity did not vary over one percent between adjoining

macro-domains the agreement of the outer boundary values was still better than one percent.

An approximate procedure to take into account the variations in c immediately beyond the boundaries of a macro-domain was investigated. This involved using the solution to the differential equation (185) to improve the value of $\hat{Z}(x)$ to be applied at the boundary micro-domain. Since only the nearest neighbor micro-domains should effect the local impedance on the boundary, the impedance at the micro-domain two micro-domains into the adjacent macro-domain normal to the boundary was assumed to have the value

$$\hat{Z}(2 \Delta x) = \sqrt{\rho \kappa (2 \Delta x)} \quad (190)$$

Equation (185) was then numerically integrated back to the boundary to obtain $\hat{Z}(0)$, (the improved value to use for the ratio of $\hat{P}(\text{bnd})/\hat{V}(\text{bnd})$). With this added refinement the boundary values again were in agreement to three significant figures for variations in c of as much as five percent between adjacent macro-domains. Variations in c as large as five percent seldom occur over macro-domains as small as considered.

1. Stability Of The Internal Energy Criterion

The criterion used for steady state determination in a macro-domain was the total amount of internal energy contained in the macro-domain. The kinetic energy of a micro-domain is given by (at any time state r),

$$T_{\text{micro}} = \frac{1}{2} \rho \Delta x \Delta y \Delta z \left[v_x^2(l, m, n, r) + v_y^2(l, m, n, r) + v_z^2(l, m, n, r) \right] \quad (191)$$

and the potential energy is

$$V_{\text{micro}}(\ell, m, n, r) = \frac{\Delta x \Delta y \Delta z}{2\kappa(\ell, m, n)} p^2(\ell, m, n, r) \quad (192)$$

Hence the macro-domain has the energy

$$U_{\text{macro}} = \sum_{\ell, m, n} (T_{\text{micro}} + V_{\text{micro}}) \quad (193)$$

For very low frequencies the function U_{macro} varies instantaneously but periodically with the time at twice the frequency of lowest harmonic present in the medium. Thus the criterion for steady state used was to monitor the instantaneous stored energy every period of the lowest frequency being simulated. This value was found to be stable to five decimal places.

2. Transient Behavior Of the Models

The behavior of the various integration schemes to a step input is illustrated by the simple model (72,73). In Appendix B is shown the step input pressure spacial distribution response of a 100 element one-dimensional domain of unit length ($\Delta x=0.01$). The responses are shown for $t=0., 0.2, 0.4, 0.6, 0.8, 1.0, 1.2, 1.4$, and 6 seconds. The speed c was taken to be unity. The symmetric form of the equations, (20,21) were programmed so that the characteristic impedance was unity. The results show the impulse excited transient oscillations characteristics of such models; however the matched condition is seen to produce no distinguishable reflections and predicted velocity of unit/sec is quite accurately simulated. The internal energy reached steady state to five decimal places in three seconds.

In Appendix C is shown the result of applying a unit amplitude cosine wave of frequency 1 Hz to the same line as in Appendix B. Again the characteristic transient oscillations occur along with some minor waveform distortion in the region of the maxima and minima of the cosine wave. These oscillations are caused by the shock excitation of the micro-domain self resonant frequency. The waveform distortions persist for a considerable period of time even after the energy has stabilized on the model. This emphasizes that caution is necessary in deciding when a model has reached steady state. In the problem shown an FFT of the spacial distribution could be used to indicate that only the desired simulation frequency is present but for three dimensional problems such a procedure is probably computationally too extravagant.

3. Two Dimensional Model

An example of a two-dimensional model integration with variable velocity c as a function of one of the dimensions is shown in Appendix D. The program used was WV2525, (see program listing). An initial condition of pressure equal to unity for $x=0$ and y varying from 0 to $25 \Delta y$ was applied. The velocity for $y=25 \Delta y$ was three times the velocity at $y=0$. For stability the integration step size required was very small, $\Delta t=10^{-4}$ sec. The two-dimensional continuous-discrete system equivalent to (68,69) was used.

In Appendix E the small amplitude transient response of the plucked membrane problem is shown. This is an example of a non-zero initial condition natural response simulation.

V. APPROXIMATE FAST FOURIER TRANSFORM COEFFICIENTS

A. INTRODUCTION AND DEFINITION OF THE TRANSFORM

The Fast-Fourier-Transform (FFT) algorithm provides an efficient digital computer algorithm for performing the one-to-one discrete sample space mapping,

$$G(k) = \frac{1}{N} \sum_{n=0}^{N-1} g(n \Delta t) e^{-j \frac{2\pi (n \Delta t)}{N \Delta t} k} \quad (194)$$

$$g(n \Delta t) = \sum_{k=0}^{N-1} G(k) e^{j \frac{2\pi k}{N \Delta t} (n \Delta t)}, \quad (195)$$

where $g(n \Delta t)$, $G(k)$ are the object and image sample spaces and Δt is the sampling interval. The properties of these transform pairs have been extensively discussed in the literature.^[19,26,27] For monochromatic signal sequences $g(n \Delta t)$, the spectral function $G(k)$ can be guessed or arrived at by heuristic reasoning but for general functions the summation in (194) can usually only be carried out analytically provided the particular finite series have known sum forms.

The continuous function Fourier series coefficients C_n where

$$g(t) = \sum_{n=-\infty}^{\infty} C_n e^{j \frac{2\pi n}{T} t} \quad (196)$$

can be obtained either by integration which is the continuous process analogous to the sum (194) or by means of symbolic differentiation^[6] using impulse functions.^[28] The possibility of using the symbolic method - modified for use on discrete sample spaces - is the chief result of this chapter.

B. REQUIREMENTS TO OBTAIN COEFFICIENTS BY THE IMPULSE METHOD

In the case of continuous functions the requirements for applicability of the impulse method of finding Fourier series coefficients are:

- (A) A spectral representation for the Dirac function $\delta(t)$.
- (B) An analytic expression relating the derivative Fourier coefficients to the original function coefficients.
- (C) A piece-wise analytic expression for the function to be represented.
- (D) A theorem giving the derivative for discontinuous functions.

It will now be shown that when requirements analogous to the above are satisfied for discrete functions, the FFT coefficients $G(k)$ can be obtained.

C. SPECTRAL REPRESENTATION FOR THE SYMBOLIC IMPULSE FUNCTION

The symbolic impulse function for discrete sample spaces is the Kronecker delta; it is defined here as the mapping function $\delta_N(s-l)$

$$\sum_{s=0}^{N-1} \delta_N(s-l) f(s) = f(l) \quad (197)$$

where $f(s)$ is a suitable testing function defined on the discrete set of points $s=0, \dots, N-1$ and is periodic modulo N ; i.e.,

$$f(s) = f(s \pm N) \quad , \quad \text{all } s \quad . \quad (198)$$

This definition is equivalent to the usual definition of the Kronecker delta as

$$\begin{aligned} \delta_N(s-l) &= 1 \quad , \quad s = l, \\ &= 0 \quad , \quad s \neq l. \end{aligned} \quad (199)$$

The function possesses the FFT coefficients

$$G(k) = \frac{1}{N} e^{-j \frac{2\pi k}{N} \ell} \quad (200)$$

as can be seen by direct application of (197) and (194). Hence

$$\delta_N(s-\ell) = \sum_{k=0}^{N-1} e^{j \frac{2\pi k}{N} (s-\ell)} \quad (201)$$

D. DEFINITION OF THE DISCRETE DERIVATIVE AND ITS SPECTRAL REPRESENTATION

Many choices are possible for defining the discrete derivative of a discrete function; the one used here is the backwards derivative commonly defined in the literature, [9]

$$g'(n\Delta t) = \frac{g(n\Delta t) - g((n-1)\Delta t)}{\Delta t} \quad (202)$$

It is a linear definition possessing the inverse

$$g(n\Delta t) = g((n-1)\Delta t) + g'(n\Delta t)\Delta t, \quad (203)$$

or predictor

$$g((n+\ell)\Delta t) = g((n-1)\Delta t) + \sum_{s=n}^{n+\ell} g'(s\Delta t)\Delta t, \quad (204)$$

where $0 \leq n \leq N-1$. Using this definition in conjunction with (195)

results in the derivative spectral representation

$$g'(n\Delta t) = \sum_{k=0}^{N-1} G(k) \frac{\{1 - e^{-j \frac{2\pi k}{N}}\}}{\Delta t} e^{j \frac{2\pi k}{N} \Delta t} (n\Delta t) \quad (205)$$

or in general for the m th derivative

$$g^{(m)}(n\Delta t) = \sum_{k=0}^{N-1} G(k) \left\{ \frac{1 - e^{-j \frac{2\pi k}{N}}}{\Delta t} \right\}^m e^{j \frac{2\pi k}{N} \Delta t} (n\Delta t) \quad (206)$$

An important result is obtained by applying (206) to (200)

$$\delta_N^{(m)}(s-l) \Delta t = \sum_{k=0}^{N-1} \left\{ \frac{1-e^{-j \frac{2\pi k}{N} \Delta t}}{\Delta t} \right\}^m e^{j \frac{2\pi k}{N} (s-l) \Delta t} \quad (207)$$

E. ANALYTIC ENVELOPE EXPRESSIONS FOR DISCRETE FUNCTIONS

An analytic-envelope representation for $g(n\Delta t)$ is necessary in order to be able to carry out discrete derivatives according to definition (202). Typical examples of such representation are

$$(A) \quad g_1(n\Delta t) = n\Delta t, \quad 0 \leq n \leq N-1$$

$$(B) \quad g_2(n\Delta t) = (n\Delta t)^2, \quad 0 \leq n \leq N-1$$

$$(C) \quad g_3(n\Delta t) = n\Delta t, \quad 0 \leq n \leq (N/2) - 1$$

$$= -n\Delta t, \quad N/2 \leq n \leq N-1$$

$$(D) \quad g_4(n\Delta t) = \sin\left(\frac{2\pi}{N\Delta t} (n\Delta t)\right), \quad 0 \leq n \leq N-1.$$

The functions (A) and (B) are polynomial type functions with no "discontinuities in analytic-envelope" over the entire interval $0 \leq n \leq N-1$. (C) is an example of a function described by two different envelope representations within the interval $0 \leq n \leq N-1$; it is said to have a discontinuity in analytic envelope in the region of $n = N/2$. (D) is an example of a transcendental function representation.

F. THEOREM ON DISCRETE DERIVATIVES OF DISCONTINUOUS ANALYTIC ENVELOPE FUNCTIONS

Analogously to continuous function symbolic theory, the unit step function $U_{s_N}(s-l)$, of a periodic sequence of discrete sample points $s=0, 1, \dots, N-1$ is defined as the linear functional mapping

$$\sum_{s=0}^{N-1} U_{sN}(s-\ell) f(s) = \sum_{s=\ell}^{N-1} f(s) . \quad (208)$$

From (194,195) it is clear that the results for the coefficients are independent of Δt ; it thus will be dispensed with in subsequent analyses.

$U_{sN}(s-\ell)$ as defined in (208) is equivalent to the piece-wise definition

$$\begin{aligned} U_{sN}(s-\ell) &= 1 , & s \geq \ell \\ &= 0 , & s < \ell . \end{aligned} \quad (209)$$

This function possesses the property that

$$\begin{aligned} \sum_{s=\ell}^{N-1} U'_{sN}(s-\ell) f(s) &= \sum_{s=0}^{N-1} \{U_{sN}(s-\ell) - U_{sN}(s-1-\ell)\} f(s) \\ &= \left\{ \sum_{s=\ell}^{N-1} - \sum_{s=\ell+1}^{N-1} \right\} f(s) \\ &= f(\ell) \\ &= \sum_{s=0}^{N-1} \delta_N(s-\ell) f(s) , \end{aligned} \quad (210)$$

so that symbolically

$$U'_{sN}(s-\ell) = \delta_N(s-\ell) \quad (211)$$

A change in a continuous analytic-envelope function can thus be described by

$$g(n) = g_1(n) + \alpha U_{sN}(n-\ell) \quad (212)$$

where $g_1(n)$ has the same analytic-envelope representation for all n . This represents the situation of a discontinuity in analytic -envelope at the point $n=\ell$. See Fig. 5. The discontinuity is a jump of magnitude α .

If (211) is used in (212) the result for the discrete derivative is

$$g'(n) = g'_1(n) + \alpha \delta_N(n-\ell) . \quad (213)$$

This proves the theorem:

Theorem: The discrete derivative of a discrete function possessing a discontinuity in analytic-envelope at the point $n=\ell$, is the backwards derivative defined in (202) for all $n \neq \ell$ plus a Kronecker delta function at $n=\ell$ of amplitude α , where

$$\alpha = g(\ell) - g_1(\ell-1) . \quad (214)$$

G. EXAMPLES

1. Coefficients of a Sawtooth Waveform

The preceding results and theorem can now be applied to a simple problem to illustrate the method. Consider the periodic sawtooth waveform shown in Fig. 6 (a). The function has the representation

$$\begin{aligned} g(n) &= n , \quad 0 \leq n \leq N-1 \\ g(n \pm N) &= g(n) . \end{aligned} \quad (215)$$

Assuming $g(n)$ has an FFT then

$$\begin{aligned} g(n) &= \sum_{k=0}^{N-1} G(k) e^{j \frac{2\pi k}{N} n} \\ &= n , \quad 0 \leq n \leq N-1. \end{aligned} \quad (216)$$

If (210) is formally differentiated by means of (196) there results , (see Fig. 6 (b))

$$\begin{aligned} g'(n) &= n - (n-1) \\ &= 1 , \quad 1 \leq n \leq N-1 \end{aligned} \quad (217)$$

and

$$\begin{aligned} g'(n) &= 0 - (N-1) \\ &= -(N-1) \quad , \quad n=0 \end{aligned} \quad (218)$$

The results (211) and (212) can be combined into the single result

$$g'(n) = 1 - N \delta_N(n) \quad , \quad 0 \leq n \leq N-1 \quad (219)$$

where now the "1" in (219) is again a continuous-envelope discrete function holding for all n , $0 \leq n \leq N-1$. In general, in applying the method, since continued differentiations of a discrete function tend to introduce more and more discontinuities in the higher order derivatives, adjustments such as performed in (219) greatly reduce the "book-keeping" required.

If now (219) is differentiated a second time the result is simply

$$g''(n) = 0 - N \delta'_N(n) \quad , \quad 0 \leq n \leq N-1. \quad (220)$$

Using (206) and (207) gives for (220) in terms of FFT expansions

$$\begin{aligned} \sum_{k=0}^{N-1} G(k) \left\{ 1 - e^{-j \frac{2\pi k}{N}} \right\}^2 e^{j \frac{2\pi k}{N} n} \\ = \frac{1}{N} \sum_{k=0}^{N-1} \left[-N(1 - e^{-j \frac{2\pi k}{N}}) \right] e^{j \frac{2\pi k}{N} n} \end{aligned} \quad (221)$$

(221) must be true for arbitrary n which requires that the coefficients of $\exp(2\pi kn/N)$ vanish for all $k \neq 0$; hence

$$G(k) = - \frac{1}{\left\{ 1 - e^{-j \frac{2\pi k}{N}} \right\}} \quad , \quad k = 1, 2, 3, \dots, N-1. \quad (222)$$

The coefficient for $k = 0$ cannot be obtained from (221) since the

bracketed term vanishes; this is to be expected since $k=0$ represents the constant term of the expansion and this must vanish upon differentiation. The $k=0$ term can be however, trivially obtained as the mean value of the function over the interval, i.e., from (194) with $k=0$,

$$G(0) = \frac{1}{N} \sum_{n=0}^{N-1} g(n) . \quad (223)$$

The results given in (222) and (223) have been checked using several FFT^[29] programs to at least six decimal places.

The procedure for the use of the method is now clear; if the discrete analytic-envelope function is polynomial in form it is evident that continued application of (202) must eventually result in the vanishing of the discrete analytic-envelope portion of the function leaving only a series of Kronecker deltas of various orders whose FFT coefficients can be written down by inspection.

2. Coefficient of Recursive Waveforms -- The Exponential Function

The question of how to treat infinite expansion polynomial analytic-envelope functions is also of interest. This would include functions such as the sine, exponential, etc.

As an example of the the treatment of these functions consider the problem of finding $G(k)$ when

$$g(n) = e^{-na} , \quad 0 \leq n \leq N-1 \quad (224)$$

$g(n)$ being periodic (see Fig. 7)

$$g(n \pm N) = g(n) . \quad (225)$$

Applying the derivative theorem gives

$$\begin{aligned} g'(n) &= e^{-na} - e^{-(n-1)a} , \quad 1 \leq n \leq N-1 \\ &= (1 - e^{-(N-1)a}) \delta_N(n) , \quad n = 0 . \end{aligned} \quad (226)$$

This can be expressed for all n in the interval as

$$\begin{aligned} g'(n) &= e^{-na} - e^{-(n-1)a} + (1 - e^{-(N-1)a}) \delta_N(n) - (1 - e^a) \delta_N(n) \\ &= e^{-na} (1 - e^a) + (e^a - e^{-(N-1)a}) \delta_N(n) , \end{aligned} \quad (227)$$

or

$$g'(n) = g(n) (1 - e^a) + e^a (1 - e^{-Na}) \delta_N(n) , \quad 0 \leq n \leq N-1 . \quad (228)$$

(228) is typical of the behavior of non-finite expansion polynomials for continuous functions: the derivative of such functions can be written in terms of the original function or certain linear combinations of the previous derivatives^[28] The analogous result can be verified to hold true for discrete functions to first order; i.e., since the derivative in (202) includes only linear difference terms and ignores in (203) contributions to $g(n \Delta t)$ proportional to $(\Delta t)^2$, $(\Delta t)^3$, etc., the general expansion for the m -th derivative $g^{(m)}(n)$

$$g^{(m)}(n) = h_0 g(n) + h_1 g^{(1)}(n) + \dots + h_{m-1} g^{(m-1)}(n) , \quad (229)$$

where h_0 , h_1 , \dots , h_{m-1} , are constants will deviate seriously from the analogous expansion in the continuous case unless Δt is small. However, though the form of the expansion (229) is different from that for continuous functions the result (229) is completely correct for discrete functions if the constant terms are correctly taken into account. Such functions are

thus linearly recursive. If (228) is now expressed in terms of the FFT the result is

$$\begin{aligned} & \sum_{k=0}^{N-1} \left\{ (1 - e^{-j\frac{2\pi k}{N}}) - (1 - e^a) \right\} G(k) e^{j\frac{2\pi k}{N} n} \\ &= \frac{1}{N} \sum_{k=0}^{N-1} \left\{ e^a (1 - e^{-Na}) \right\} e^{j\frac{2\pi k}{N} n} . \end{aligned} \quad (230)$$

This requires that

$$\begin{aligned} G(k) &= \frac{e^a (1 - e^{-Na})}{N(e^a - e^{j\frac{2\pi k}{N}})} \\ &= \frac{1 - e^{-Na}}{N(1 - e^{-(a+j\frac{2\pi k}{N})})} , \quad k=1, 2, 3, \dots, N-1 . \end{aligned} \quad (231)$$

This particular result can be checked exactly analytically, for the function (224), when substituted into (194) becomes

$$\begin{aligned} G(k) &= \frac{1}{N} \sum_{n=0}^{N-1} e^{-na} e^{-j\frac{2\pi k}{N} n} \\ &= \frac{1}{N} \sum_{n=0}^{N-1} e^{-n(a+j\frac{2\pi k}{N})} \\ &= \frac{1}{N} \sum_{n=0}^{N-1} x^n , \end{aligned} \quad (232)$$

where

$$x = e^{-(a+j\frac{2\pi k}{N})} . \quad (233)$$

Equation (232) is simply a finite geometric series with the sum

$$\begin{aligned}
 G(k) &= \frac{1}{N} \frac{1-x^{(N-1)+1}}{(1-x)} \\
 &= \frac{1}{N} \left(\frac{1-x^N}{1-x} \right) \\
 &= \frac{1-e^{-Na}}{N(1-e^{-(a+j\frac{2\pi k}{N})})}, \quad k=0,1,\dots,N-1 \quad (234)
 \end{aligned}$$

Note that in this case the term $k=0$ has been obtained along with the usual other terms; this is because in (22) the derivative has been expressed recursively in terms of the complete original function e^{-na} .

F. GENERAL PROCEDURE

The above two examples illustrate the general procedure to be followed for finding the coefficients of a Fast Fourier Transform. The procedure may be summarized and generalized as follows:

- (A) If the analytic-envelope is finite polynomial in form continually differentiate the function until only Kronecker delta functions remain. After each differentiation, adjust the analytic form so that the particular derivative holds at all points within the interval by extending the analytic-envelope over the entire domain $0 \leq n \leq N-1$ with suitable modification of the Kronecker delta amplitude.
- (B) Equate the spectral representations for $g(n)$ to obtain the coefficients for all $k \neq 0$ terms. The $k=0$ term is obtained as the average value of the discrete waveform. This can be obtained in many cases exactly by use of the finite arithmetic or geometric sum formulas.
- (C) If the analytic-envelope consists of sub-regions over the interval $0 \leq n \leq N-1$ defined by two or more different functions $g_1(n)$, $g_2(n)$, etc., the problem can be handled by multiplying the appropriate sub-region function definitions by unit step functions for the appropriate regions. Differentiation of the resulting products using the usual product rule will lead to a simplified analytic-envelope function plus the necessary Kronecker deltas. The necessary work quickly becomes prohibitive unless the simplified analytic-envelope can be easily

extended over the whole interval as done in the previous problems.

- (D) If the function is transcendental, use the recursive procedure shown in the second example.

In conclusion, the method presented in this section provides an analytic method for determining closed form formulas for FFT coefficients. These expressions can be used in theoretical discussions to study the behavior of the coefficients for various sampling intervals, size of discontinuity present, accuracy of computer programs, etc.

VI. TWO APPROACHES TO THE OPTIMUM ACOUSTIC ARRAY PROCESSOR

A. INTRODUCTION

A subject of great importance and deserving of continued study is the detection of acoustic communication signals in the presence of a background noise field.^[30] In this chapter several approaches to the solution of this problem are reviewed and the results of simulating one of the approaches on a computer are reported and discussed.

The usual means of detecting and receiving acoustic signals, corrupted by noise is by a spacially-distributed array of acoustic sensors. Such arrays serve the fundamental dual purposes of providing multiplication of individual sensor gain and spacial correlation. A typical array of sensors together with the associated signal processing network is shown in Fig. 8.

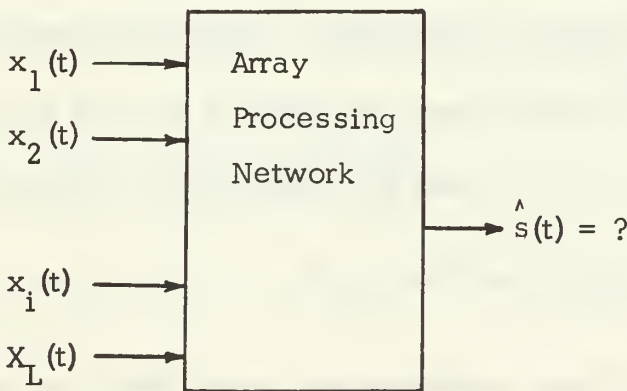


Fig. 8. Array with associated processing network.

Although analog, digital, or a combination of techniques may be involved in the signal processing network in Fig. 8, a few qualitative

generalizations can be made. If the input signal $s(t)$ is wideband and quite high information rates are involved the decision to "go analog" is almost certain. The required computation rates for real time processing are very large in such cases even taking into account the possibility of using such tools as the FFT algorithm; thus digital processing is restricted to relatively low frequencies. However, the FFT algorithm, coupled with very high speed special purpose computers, have made possible data rates of 10 kHz and higher so that in the near future the use of digital real-time parallel processing may be expected to increase considerably. [31]

There are many different approaches to the problem of optimum array processing; two main approaches will be discussed here. The first approach discusses the possibility of modifying classical analog optimum quadratic-cost-function detection and estimation theory to make use of the FFT algorithm; the second approach considers the use of Widrow adaptive array processor and possible modifications.

B. THE MODIFIED OPTIMUM DETECTION - ESTIMATION ARRAY PROCESSOR

Following Cox^[32] closely let it be assumed that the L sensors in Fig. 8 have inputs $x_i(t)$, $i=1, L$ and that the i -th input can be written as

$$x_i(t) = \int_{-\infty}^{\infty} m_i(t-\tau)s(\tau) d\tau + n_i(t) \quad , \quad i=1,2,\dots,L \quad (235)$$

where $m_i(t)$ is a known linear transformation which takes into account the i -th time delay of arrival of the scalar signal $s(t)$. $m_i(t)$ thus provides the generality to account for local dispersive properties of the medium, non-uniform array spacing, and phase distortion of particular sensors. The

function $\underline{n}_1(t)$ is assumed to be additive medium noise. The relation (235) can be re-written as the vector equation

$$\underline{x}(t) = \int_{-\infty}^{\infty} \underline{m}(t-\tau) s(\tau) d\tau + \underline{n}(t) \quad (236)$$

where now \underline{x} , \underline{m} , and \underline{n} and L -dimensional column vectors. Several investigators have formed large hyper-vector array ensembles of the quantities \underline{x} , \underline{m} , and \underline{n} in the form of correlation matrices at various times t_1, t_2 , etc., in order to then use optimum estimation techniques [30,31]; these procedures without the assumption of stationarity are confined to very low frequencies because of the great magnitude of the computation requirements. [33,34]

The assumption of stationarity will be made in the following sections in order to apply frequency domain techniques. Let it be assumed that

$$\begin{aligned} \underline{x}(t) &= \sum_{k=-\infty}^{\infty} \underline{\psi}'(k) e^{j2\pi k f_1 t}, \quad s(t) = \sum_{k=-\infty}^{\infty} \sigma'(k) e^{j2\pi k f_1 t} \\ \underline{n}(t) &= \sum_{k=-\infty}^{\infty} \underline{\eta}'(k) e^{j2\pi k f_1 t} \end{aligned} \quad (237)$$

$$\underline{m}(t) = \int_{-\infty}^{\infty} \underline{\mu}'(f) e^{j2\pi f t} df, \quad (238)$$

where

$$f_1 = \frac{1}{T}. \quad (239)$$

Then, substituting in (236) requires that

$$\begin{aligned} \underline{0} &= \sum_{k=-\infty}^{\infty} \left\{ (\underline{\psi}'(k) - \underline{\eta}'(k)) e^{j2\pi k f_1 t} - \int_{-\infty}^{\infty} \underline{\mu}'(f) \sigma'(k) e^{j2\pi f t} \right. \\ &\quad \left. \cdot \int_{-\infty}^{\infty} d\tau e^{j2\pi \tau (k f_1 - f)} \right\} \end{aligned}$$

$$\begin{aligned}
&= \sum_{k=1}^{\infty} \left\{ (\underline{\psi}'(k) - \underline{n}'(k)) e^{j2\pi k f_1 t} - \int_{-\infty}^{\infty} \underline{\mu}'(f) \sigma'(k) e^{j2\pi f t} \delta(f - f_1 k) df \right\} \\
&= \sum_{k=1}^{\infty} \left\{ \underline{\psi}'(k) - \underline{\mu}'(k) \sigma'(k) - \underline{n}'(k) \right\} e^{j2\pi f_1 k t} .
\end{aligned} \tag{240}$$

This can vanish identically for arbitrary t if and only if the bracket term vanishes which gives

$$\underline{\psi}'(k) = \underline{\mu}'(k f_1) \sigma'(k) + \underline{n}'(k) , \quad \text{all } k . \tag{241}$$

(241) suggest arranging the equation in the matrix forms

$$\underline{\psi} = \begin{pmatrix} \underline{\psi}'(1) \\ \underline{\psi}'(2) \\ \vdots \end{pmatrix} , \quad \underline{n} = \begin{pmatrix} \underline{n}'(1) \\ \underline{n}'(2) \\ \vdots \end{pmatrix} , \quad \sigma = \begin{pmatrix} \sigma'(1) \\ \sigma'(2) \\ \vdots \end{pmatrix} \tag{242}$$

$$\begin{aligned}
\underline{\mu} &= \begin{pmatrix} \underline{\mu}'(f_1) & \underline{0} & \underline{0} & \dots \\ \underline{0} & \underline{\mu}'(2f_1) & \underline{0} & \\ \vdots & \vdots & \underline{\mu}'(3f_1) & \ddots \end{pmatrix} \\
&= \text{diag} (\underline{\mu}'(f_1), \underline{\mu}'(2f_1), \dots)
\end{aligned} \tag{243}$$

or

$$\underline{\psi} = \underline{\mu} \sigma + \underline{n} \tag{244}$$

where $\underline{\psi}'(1)$ represents the Fourier series coefficients of the entire array of L elements at frequency f_1 , $\underline{\psi}'(2)$ the Fourier coefficients of the entire array at frequency $2f_1$, etc. If the signal $\underline{s}(t)$, input vector $\underline{x}(t)$ and noise $\underline{n}(t)$ are not band-limited the number of rows of the equation (244) is infinite; the usual assumption is that $\underline{x}(t)$ is band-limited to some upper frequency f_{\max} where

$$\begin{aligned}
 f_{\max} &= k_{\max} f_1 \\
 &= \frac{k_{\max}}{T} ,
 \end{aligned}
 \tag{245}$$

and T is the observation period.

Denoting the noise auto-spectrum at frequency $k f_1$ by $\underline{R}(k)$ where

$$\underline{R}(k) = E \left\{ \underline{\eta}(k) \underline{\eta}^*(k) \right\} , \tag{264}$$

and assuming that the period of observation T is large compared to the maximum lag for which the auto-correlation function is significantly different from zero, the cross spectrum $\underline{R}(k, \ell)$ satisfies

$$\begin{aligned}
 \underline{R}(k, \ell) &= E \left\{ \underline{\eta}(k) \underline{\eta}^*(\ell) \right\} \\
 &= \underline{0} , \quad \text{all } k \neq \ell ;
 \end{aligned}
 \tag{247}$$

the noise covariance matrix is thus of the form

$$\begin{aligned}
 \underline{R} &= \begin{bmatrix} \underline{R}(1) & \underline{0} & \underline{0} & \dots \\ \underline{0} & \underline{R}(2) & \underline{0} & \\ \underline{0} & \underline{0} & \underline{R}(3) & \\ \vdots & & & \ddots \end{bmatrix} \\
 &= \text{diag} (\underline{R}(1), \underline{R}(2), \underline{R}(3), \dots) .
 \end{aligned}
 \tag{248}$$

In (246) $\underline{\eta}^*$ is the complex conjugate transpose of $\underline{\eta}$. The matrix \underline{R} is square and each submatrix on the diagonal is of dimension $L \times L$.

Suppose $s(t)$ is assumed random with zero mean; a spectral covariance matrix \underline{P} where

$$\underline{P} = E \left\{ \sigma(k) \sigma^*(\ell) \right\} , \tag{249}$$

can be defined. Under the assumption again that the expansion period T is long, \underline{P} is also diagonal having the form

$$\underline{P} = \begin{bmatrix} p_1 & 0 & 0 & \dots \\ 0 & p_2 & 0 & \\ 0 & 0 & p_3 & . \\ \vdots & & & . \end{bmatrix}$$

$$= \text{diag} (p_1, p_2, p_3 \dots) \quad , \quad (250)$$

where p_k is the power in the spectral line kf_1 . Because of diagonal forms of the matrices $\underline{\mu}$, \underline{P} , and \underline{R} , the test statistic for the detection problem, ζ , which when compared with a threshold statistic determines whether a signal has occurred or not, can be expressed in the simple form

$$\zeta = \frac{1}{2} \sum_{k=1}^k \left| a(k) (\underline{\mu}'(k))^* (\underline{R}(k))^{-1} \underline{\psi}'(k) \right|^2 \quad , \quad (251)$$

where

$$|a(k)|^2 = \frac{p_k}{1 + p_k (\underline{\mu}'(k))^* (\underline{R}(k))^{-1} \underline{\mu}'(k)} \quad . \quad (252)$$

The form (251) is simply a sum of terms over the spectrum of the signal and is derived as the ratio of the conditional probabilities

$$\zeta = \frac{p(\underline{\psi} / (\underline{\mu} \sigma + \underline{\eta}))}{p(\underline{\psi}' / \underline{\eta})} \quad , \quad (253)$$

where

$p(\underline{\psi} / (\underline{\mu} \sigma + \underline{\eta}))$ = probability that during an observation interval T the input spectra contains the signal plus noise spectra,

$p(\underline{\psi} / \underline{\eta})$ = probability that during an observation interval T the input spectra contains only noise spectra.

If now a vector spectral function $\underline{\theta}^*(k)$,

$$\underline{\theta}^*(k) = \underline{\mu}'^*(k) (\underline{R}(k))^{-1} \quad (254)$$

and a scalar spectral response function $\Omega(k)$,

$$\Omega(k) = a(k) \underline{\theta}^*(k) \underline{\psi}(k), \quad (255)$$

are defined Parseval's theorem gives the simple result for $\zeta(T)$

$$\zeta(T) = \frac{1}{2T} \int_0^T \left| \sum_{-k_{\max}}^{k_{\max}} \Omega(k) e^{j2\pi k f_1 t} \right|^2 dt. \quad (256)$$

Equation (256) in practice can be accomplished by averaging the square law detected output of the spectral function $\Omega(k)$. The function $\underline{\theta}^*(k)$ performs the operations of pre-whitening the input signal and then matching to the assumed known spacial structure of the signal. Fig. 9 shows the complete system; it is composed of a system of weighted bandpass filters on each sensor followed by spacial spectral summation, multiplication, square law detection and averaging.

This processor can be generalized to supply with but slight modification not only detection and optimum estimation of a random signal $\hat{s}(t)$ but also detection and optimal estimation for the known and unknown signal problems. Figure 10 shows the general all-purpose array processor assuming analog processing.

The use of an analog spectrum processor has several disadvantages even though high real-time data rates are possible. Particularly objectionable is the fact that the banks of narrow band-pass "comb" filters required at the outputs of each sensor are difficult to maintain in precise adjustment track between sensors and are quite difficult to shift in frequency. In addition the process of analog multiplication is rather crude.

As an alternative to the analog spectrum processor, the possibility of using the discrete FFT algorithm to perform the spectral analysis was considered. The essential modifications in the above analysis are as follows.

Consider equations (237,238). Assuming stationarity the quantities $\underline{x}(t)$, $\underline{n}(t)$ and $s(t)$ could be discretized by a digital sampler at the output of each sensor to provide the data values $\underline{x}(i \Delta t)$, $\underline{n}(i \Delta t)$, $s(i \Delta t)$ where $0 \leq i \leq (N-1)$ and the sampling interval Δt in any observation interval T satisfies

$$N \Delta t = T .$$

Since the maximum harmonic that can be represented by the finite discrete N point Fourier transform is

$$\begin{aligned} f_{\max} &= \left(\frac{N}{2} - 1 \right) f_1 \\ &= \frac{\left(\frac{N}{2} - 1 \right)}{T} , \end{aligned}$$

the output of each sensor must be low-pass filtered to exclude all components of frequency greater than f_{\max} if the finite band-width of the sensor has not already provided the needed filtering. Failure to include such filtering will cause aliasing. [26] The discrete equivalent of (237) is thus

$$\begin{aligned} \underline{x}(\xi) &= \sum_{k=0}^{N-1} \underline{\psi}'(k) e^{j \frac{2\pi k}{N} \xi} , \quad \underline{n}(\xi) = \sum_{k=0}^{N-1} \underline{\eta}'(k) e^{j \frac{2\pi k}{N} \xi} , \\ (\xi) &= \sum_{k=0}^{N-1} \sigma(k) e^{j \frac{2\pi k}{N} \xi} , \quad \xi = 0, 1, \dots, N-1. \end{aligned} \quad (257)$$

In (238) the spacial weighting vector $\underline{m}(t)$ is represented by a bi-lateral infinite Fourier transform and not a Fourier series. Ignoring this important difference for the moment and assuming that $\underline{m}(\xi)$ can be represented by the finite discrete transform

$$\underline{m}(\xi) = \sum_{\ell=0}^{N-1} \underline{\mu}'(\ell) e^{j \frac{2\pi \ell}{N} \xi}, \quad \xi = 0, 1, \dots, N-1 \quad (258)$$

the discrete equation equivalent to (240) becomes

$$0 = \sum_{k=0}^{N-1} \left\{ (\underline{\psi}'(k) - \underline{\eta}'(k)) e^{j \frac{2\pi k}{N} \xi} - \sum_{\tau=0}^{N-1} \sum_{\ell=0}^{N-1} \underline{\mu}'(\ell) e^{j \frac{2\pi \ell}{N} (\xi - \tau)} \cdot \sigma(k) e^{j \frac{2\pi k}{N} \tau} \right\}$$

$$= \sum_{k=0}^{N-1} \left\{ (\underline{\psi}'(k) - \underline{\eta}'(k)) e^{j \frac{2\pi k}{N} \xi} - \sum_{\ell=0}^{N-1} \underline{\mu}'(\ell) e^{j \frac{2\pi \ell}{N} \xi} \sigma'(k) \sum_{\tau=0}^{N-1} e^{j \frac{2\pi \tau}{N} (k - \ell)} \right\}$$

$$= \sum_{k=0}^{N-1} \left\{ (\underline{\psi}'(k) - \underline{\eta}'(k)) e^{j \frac{2\pi k}{N} \xi} - \sum_{\ell=0}^{N-1} \underline{\mu}'(\ell) e^{j \frac{2\pi \ell}{N} \xi} \sigma'(k) \delta_N(k - \ell) \right\}$$

$$= \sum_{k=0}^{N-1} \left\{ \underline{\psi}'(k) - \underline{\mu}'(k) \sigma'(k) - \underline{\eta}'(k) \right\} e^{j \frac{2\pi k}{N} \xi} \quad (259)$$

Employing the same argument as in (240) the result

$$\underline{\psi}'(k) = \underline{\mu}'(k) \sigma(k) + \underline{\eta}'(k) \quad , \quad k=0, 1, \dots, N-1, \quad (260)$$

follows. Thus if (258) is valid the problem is again cast into the framework of (241) and the subsequent analysis follows directly.

Now equation (238) in the analog formulation must involve a continuous frequency spectrum in order that any possible phase shift corresponding to any possible time delay $\delta(t-t_i)$ relating the response time at the i -th sensor to that at some convenient origin may be represented. The Kronecker delta in (259) does not possess such resolution; it can represent at most discrete time delays, separated by a minimum time of Δt sec. and a maximum time of $N \Delta t = T$ sec. Thus only certain discrete angular resolution is possible with the L element array under the representation restriction (238); the array has an angular boresight error of

$$\delta \phi \sim \frac{c \Delta t}{L \delta \lambda} \quad (261)$$

where c is the local acoustic velocity and $\delta \lambda$ is the fraction of a wavelength between elements for a uniform array detecting acoustic signals of wavelength λ . Hence for fine angular resolution Δt should be small. A more fundamental restriction is imposed by the maximum time delay representable i.e., $N \Delta t$. Typical high resolution acoustic arrays may be of 30λ to 50λ in aperture and in an endfire configuration this means that time delays of the order of

$$\text{delay} \sim \frac{50 \lambda}{c} \quad (262)$$

across the array are possible. As a consequence even if T were not required to be long in order to guarantee the diagonal properties of the matrices \underline{P} and \underline{R} , it would be necessary for it to be long in order to satisfy (257).

The net result of these considerations is that provided T is at least greater than some criterion such as (262) and Δt is small enough to satisfy desired angular resolution requirements such as (261) the analysis in (259) is valid and hence the FFT digital spectral analysis can be applied to optimally detect and estimate in the least squares sense the presence of a signal $s(t)$ in the presence of background noise.

C. ADAPTIVE ARRAYS

1. The Widrow Scheme

The approach outlined above is classical in nature and relies on the statistical properties of signals alone to perform detection and estimation. The procedure requires the inversion of the noise covariance matrix which involves considerable computation and can only be completely avoided if the noise is known a priori to be white. Further, apart from increased gain (which in general amplifies both the signal and noise) no particular correlation properties of the array are utilized.

Conceptually, as previously stated in the introduction, an ideal array processor should be able to adapt to both the specific spacial-location medium properties where the array is to be placed and an incoming signal.

Widrow^[40] and his co-workers have devised an adaptive scheme which appears to have some of these properties as well as other advantages

over the classical procedure in those cases in which the statistics of the signal $s(t)$ are known and the noise field has non-uniform directional properties. No particular properties of the noise field are required except that the incident noise direction be different from that of the desired incident signal. The initial development given below is considerably expanded over that included in Widrow's paper, being a generalization to the vector case of a paper by Mantey. [41]

In Fig. 11 is shown a discrete sampled-data network involving time delay networks for the i -th array element shown in Fig. 8. From Fig. 11, the output $y_i(k)$ can be written as a function of the previous inputs and outputs as

$$y_i(k) = a_1 y_i(k-1) + a_2 y_i(k-2) + \dots + a_N y_i(k-N) + b_0 x_i(k) + b_1 x_i(k-1) + \dots + b_M x_i(k-M) , \quad (263)$$

where $x_i(k)$ is the input sample at time $k\Delta t$, $x_i(k-1)$ is the input sample at time $(k-1)\Delta t$, $y_i(k)$ is the output of the network at time $k\Delta t$, $y_i(k-1)$ is the output of the network at time $(k-1)\Delta t$, etc. The coefficients a, \dots, a_N are referred to as feedback coefficients and the coefficients b_0, \dots, b_M are called feedforward coefficients. The term feedback here comes from the fact that the output $y_i(k)$ is determined not only by the present and past inputs but also by the past outputs of the network. Such networks thus possess "memory" in that, indirectly, the output $y_i(k)$ depends on all previous inputs $x_i(k)$ of the network. This is a highly desirable property. In detection theory the use of the matched filters to estimate threshold levels optimally in the least mean squares sense is

important^[32]; such a filter is required to have an impulse response which is the convolution of the known signal waveform which is to be detected. Thus if the network in Fig. 11 is required to simulate such a filter it is necessary that its impulse response be as close to the waveform assumed a priori to be contained in $x_i(t)$ as possible. An important consideration in this regard is the relative efficiency in providing memory of each of the terms in (263); the impulse response is non-vanishing for the purely feedforward terms b_0, b_1, \dots, b_M , only over the intervals $(0 \leq k \Delta t \leq M \Delta t)$ whereas the feedback coefficients provide semi-infinite non-vanishing response, $(0 \leq k \Delta t \leq \infty)$. Thus a much simpler feedback network can provide theoretically a sampled impulse response which can be made more nearly a "match" for the desired signal. As pointed out by Mantey^[57] the cost in computing time and not computer storage is the essential limitation on the real time application of digital signal processing schemes; it can be seen from (263) that each feedback section "costs" the same as each feed-forward section in computing time. The sample data network containing a single input $b_0 x(k)$, and N feedback terms $a_N y(k-N), \dots, a_1 y(k)$ is called an auto-regressive network and has been extensively studied by Whittle^[42] and Claerbout,^[43] who conclude that the network has apparent advantages over other schemes in that it appears to yield more physically realizable models for stochastic processes which are time variable.

Referring to Fig. 11 and equation(263) it is clear that the output of the entire L element array can be written in vector form as

$$\underline{y}(k) = \sum_{n=1}^N A_n \underline{y}(k-n) + \sum_{m=0}^M B_m \underline{x}(k-m) \quad (264)$$

where

$$\underline{y}(k) = \begin{pmatrix} y_1(k) \\ \vdots \\ y_L(k) \end{pmatrix}, \quad \underline{x}(k) = \begin{pmatrix} x_1(k) \\ \vdots \\ x_L(k) \end{pmatrix} \quad (265)$$

$$A_n = \text{diag}(a_{1n}, a_{2n}, \dots, a_{Ln}), \quad B_m = \text{diag}(b_{1m}, b_{2m}, \dots, b_{2m}). \quad (266)$$

In (264) there are N $L \times L$ matrices A_n and $M+1$ $L \times L$ matrices B_m .

The theory of sample-data systems may be most elegantly treated by the Z transform^[44], which is defined in terms of a complex variable z and a uniform sample sequence $f(k)$ as

$$\begin{aligned} F(z) &= \sum_{k=0}^{\infty} f(k) z^{-k} \\ &= Z(f(k)) \end{aligned} \quad (267)$$

and has the property that the transform of a unit pulse delayed ℓ samples in time $\delta_{k,\ell}$, is

$$Z(\delta_{k,\ell}) = z^{-\ell}. \quad (268)$$

Applying the transform to the system (264) thus gives

$$\begin{aligned} \underline{Y}(z) &= \sum_{n=1}^N A_n Z(\underline{y}(k-n)) + \sum_{m=0}^M B_m Z(\underline{x}(k-m)) \\ &= \sum_{n=1}^N A_n z^{-n} \underline{Y}(z) + \sum_{m=0}^M B_m z^{-m} \underline{X}(z) \\ &= \sum_{n=1}^N (A_n z^{-n}) \underline{Y}(z) + \sum_{m=0}^M (B_m z^{-m}) \underline{X}(z), \end{aligned} \quad (269)$$

where

$$\underline{Y}(z) = \mathcal{Z}(\underline{y}(k)) \quad , \quad \underline{X}(z) = \mathcal{Z}(\underline{x}(k)) \quad . \quad (270)$$

Defining the matrices A and B by

$$\begin{aligned} A &= \sum_{n=1}^N A_n z^{-n} \\ &= \sum_{n=1}^N \text{diag}(a_{1n}, a_{2n}, \dots, a_{Ln}) z^{-n} \\ &= \sum_{n=1}^N \text{diag}(a_{1n} z^{-n}, a_{2n} z^{-n}, \dots, a_{Ln} z^{-n}) \\ &= \text{diag} \left(\sum_{n=1}^N a_{1n} z^{-n}, \sum_{n=1}^N a_{2n} z^{-n}, \dots, \sum_{n=1}^N a_{Ln} z^{-n} \right), \\ B &= \sum_{m=0}^M \text{diag}(b_{1m}, b_{2m}, \dots, b_{Lm}) z^{-m} \\ &= \text{diag} \left(\sum_{m=0}^M b_{1m} z^{-m}, \sum_{m=0}^M b_{2m} z^{-m}, \dots, \sum_{m=0}^M b_{Lm} z^{-m} \right), \end{aligned} \quad (271)$$

the equation (269) becomes

$$\underline{Y}(z) = A \underline{Y}(z) + B \underline{X}(z), \quad (272)$$

which may be written as

$$\begin{aligned} \underline{Y}(z) &= (I-A)^{-1} B \underline{X}(z) \\ &= \text{diag} \left(\frac{\sum_{m=0}^M b_{1m} z^{-m}}{1 + \sum_{n=1}^N a_{1n} z^{-n}}, \dots \right) \underline{X}(z) \end{aligned} \quad (273)$$

The transfer function for the i-th sensor is thus

$$G_i(z) = \frac{Y_i(z)}{X_i(z)} = \frac{b_0 + b_{11}z^{-1} + b_{12}z^{-2} + \dots + b_{1M}z^{-M}}{1 - a_{11}z^{-1} - a_{12}z^{-2} - \dots - a_{1N}z^{-N}} \quad (274)$$

The denominator polynomial in (274) determines the stability of the network; it is shown in Ragazzini and Franklin^[44] that if the magnitude of the complex roots of the denominator polynomial are less than unity the network characterized by equation (263) is stable. Since the transfer function corresponding to (273) is diagonal and involves no cross-coupling between the individual sensor networks, the question of array stability depends only on the stability of each individual sensor; if all of the sensor networks individually are stable and have finite bounded outputs, their sum will also have finite bound and be stable. However, the problem of stability in (274) is not necessary to consider at the stage of the problem development here since the general sensor network in Fig. 10 will be considerably modified in the subsequent development.

The basic Widrow adaptive network is obtained from (274) by setting all of the feedback coefficients a_i , $i=1, \dots, N$ to zero, i.e., the network is a pure feed-forward network such as shown in Fig. 12. Fig. 12 represents a general M -th order feed-forward network and Widrow indicates that the identical delay networks shown will allow the network to receive signals over a band of frequencies if the weights b_1, \dots, b_M are properly adjusted. In Fig. 13 is shown a single stage of the network with the delay adjusted to $\pi/2$ radians for some input frequency f_1 . Since at frequency f_1 the general output would be

$$y(t) = b_0 \cos(2\pi f_1 t + \phi_1) + b_1 \sin(2\pi f_1 t + \phi_1) \quad (275)$$

it is clear that such a section could represent perfectly an input signal at frequency f_1 with arbitrary incident phase angle. Returning to Fig. 12, it is clear that M such sections in cascade will allow the exact representation of any input signal which can be written as the finite bandwidth Fourier series

$$y(t) = \sum_{m=-M}^M c_m e^{j2\pi m f_1 t} \quad (276)$$

which represents a real bandwidth of Mf_1 Hz. If the delay is not precisely $\pi/2$ radians for the lowest frequency of interest in the input signal the array can still compensate by readjustment of the weights a_1, \dots, a_M but the magnitudes of the higher order coefficients quite rapidly (in the case of many delays) become very large. More will be said relative to this point later.

Since the matrix B in (273) is diagonal it is convenient in considering the Widrow formulation to define a column weight vector \underline{w} whose components are the diagonal components for the i -th array sensor from B before the application of the Z transform,

$$\underline{w} = \begin{pmatrix} b_0 \\ \cdot \\ \cdot \\ \cdot \\ b_M \end{pmatrix} \quad (277)$$

Although the whole array of sensors could be handled corporately, it is only necessary to consider the sensors individually since there is no cross-coupling between sensors during the adaptive process to be described.

Defining a vector $\underline{x}_i(t)$ as the array of inputs to the weighting coefficients in Fig. 12, the output $y_i(t)$ can be written as

$$y_i(t) = \underline{w}^*(t) \underline{x}_i(t), \quad (279)$$

or in discrete form

$$y_i(k) = \underline{w}^*(k) \underline{x}_i(k) \quad . \quad (280)$$

A very necessary requirement for the use of Widrow's adaptive algorithm is knowledge of the signal $s(t)$, or discretely $s(k)$. If $s(k)$ is known an error signal $\epsilon(k)$, can be defined as the difference between the value $s(k)$ and the present output of the network given by (280),

$$\epsilon(k) = s(k) - \underline{w}(k) * \underline{x}_i(k). \quad (281)$$

A local gradient search technique, based upon the method of steepest descent, is then used to establish an algorithm for iterating the weight vector $\underline{w}(k)$ to a new value $\underline{w}(k+1)$; the algorithm

$$\underline{w}(k+1) = \underline{w}(k) - 2c_\nu \epsilon(k) \underline{x}_i(k) \quad (282)$$

can be shown to converge after a large number of iterations to the Wiener-Holf solution. In (282) c_ν is a proportionality constant which is always negative and usually satisfies

$$-1 \leq c_\nu \sum_{i=j}^L \underline{x}_i * \underline{x}_i < 0. \quad (283)$$

The necessity of knowing the signal $s(k)$ which is to be received is the greatest restriction of the algorithm; since the signal is not in general known, except for possibly its second order statistics an a priori assumed artificial signal $s(k)$ must be continually injected into the processor in order for it to adapt. This can be done in two ways. The

artificial signal $s(k)$ can be injected into the processor directly for an adaptive period τ_a during which it cannot be used for receiving, and then the artificial signal is turned off and the performance of the processor "decays" to degraded performance over a period τ_u during which the processor can be used for reception. The mechanism for doing this is shown in Fig. 14. If the power spectral properties of the artificially injected signal $s(k)$ closely approximate those in the incoming waveform $\underline{x}_i(k)$ the τ_u periods will be long relative to the τ_a periods. An alternative method is to provide two processors in parallel; an adaptive processor into which an artificial signal is continually fed to produce the optimum weights \underline{w} to which a second processor is slaved to have the same weights and can thus continually receive external signals. Neither approach appears to be very satisfactory; the on-off approach is clearly useable only part of the time and the continuous, two-processor approach is expensive to implement and is "prejudiced" in favor of the a priori assumed signal $s(k)$.

2. A Visual Display Program

Modifying a program, on which substantial previous programming effort had been contributed by LCDR William Moorehead of the U. S. Navy, an interactive study of the Widrow algorithm was performed. The program employs an IBM 360/67 computer with attached Model (2250-1) display unit; the flow chart for the program is shown in Fig. 15. By means of attention keys located at the display console (see Fig. 16) the program simulation could be interrupted dynamically at any point to provide additional information, change parameters, or produce permanent

Calcomp plots of any of the console displayed results.

The basic information required for input to the program is the assumed direction and spectrum (center frequency and bandwidth) of the desired signal. The desired input signal power is automatically adjusted by the program to be unity with uniform power density being spread among the various sideband components over the desired bandwidth. The actual signal is then synthesized by use of the inverse discrete FFT algorithm.

The noise field in the experiments that were performed was assumed to be incident normally to the array (see Fig. 17; as such the effective aperture of the noise signal is greater than that of incident signals for all incident signal directions not also normal to the array. Widrow employed sinusoidal "noise" for his test examples; the present investigation used random noise. The additive random Gaussian noise was added to the array from a random number generator program which produced random numbers of zero mean and known standard deviation.

The time delay taps on the delay line were variable from a minimum of every sampling interval to any larger integral multiple of the basic sampling interval.

At any given time in the adaption procedure up to four plots of directivity pattern vs. frequency and four plots of frequency response vs. angle could be obtained. This allows the receiving patterns at frequencies different from the desired frequency and frequency response diagrams for angles different from the desired look angle to be obtained

simultaneously. In addition the program provides the output spectrum of the array filter and the array filter weights $w(k)$.

In Appendix F are shown the results of running the program for the physical situation illustrated in Fig. 17.

In general the results are as follows. The filter provides excellent suppression for all frequencies outside of the band $400 < f < 600$ cps. in the desired look direction of $\theta = 45^\circ$. The response of the filter to other frequencies and angles of incidence is much reduced with the exception of frequencies above 1000 Hz where the pattern exhibits the familiar structure of linear arrays with element spacings greater than $\lambda/2$. i.e., the many lobe problem. This problem can be easily overcome, however, by either closer array element spacing or by ignoring the pattern and placing low pass filters of pass-band, $0 \leq \text{pass-band} \leq 2f_1$ where f_1 is the desired signal frequency of interest.

In addition it is clear that the adaption time for the filter to reach steady state is quite long for general white noise inputs of the type used here; much longer than for the narrow band sinusoidal noise used in Widrow's paper. [40]

3. Other Approaches: Future Research

The results reported in this paper relative to the Widrow filter simulation program are minimal; rather it is thought more appropriate to make a few generalizations and suggestions for future research based upon results already observed.

The Widrow filter must have knowledge of the signal to be detected (at least the nominal frequency and bandwidth); since a priori the signal

to be detected is of course not known an assumed signal $s(k)$ must be applied to adapt the filter. From a practical point of view this must mean a continual process of adaption using an ensemble of maximum likelihood input signals $s(k)$ as the adaptive signal inputs; the speed of adaption is thus of fundamental importance as all angles, bandwidth frequencies, and nominal center frequency ranges of the ensemble must be scanned during the acquisition phase of operation. The single-mode two-processor adaption algorithm is thus almost a necessity.

Secondly, the real time speed of adaption is strongly a function of the amount of adaption computations the processor is required to do. In this regard the alternative procedure of assuming every input is narrow-band to first order during the acquisition phase (and thus requires only the computations for b_0, b_1 for any element) appears to have special merit, especially for the detection of amplitude modulated signals. During the acquisition phase b_2, \dots, b_M would be maintained at zero. After acquisition of a signal the "fine structure" of the broadband spectrum can be provided by adding other spectral components to the adaption signal and adapting the weights $b_0, b_1, b_2, \dots, b_M$ as before. The signal output during the preliminary adaption phase would be quite distorted; this might be a tolerable price to pay for much faster speed of adaption.

Another possibility for increased speed of adaption as well as providing more stable weights would appear to be the use of non-uniform time delays. There is nothing in the recursive algorithm (276) which requires that the delays all be equal; the algorithm requires only that the

weight vector, error scalar, and particular input signal be known at any sequential time step. Consider for example the amplitude modulation case in which the desired signal is as shown in Fig. 18. This signal $s(t)$ is wide-band in the sense that the (band-width) / (center-frequency) $\sim .5$ but is narrow-band in the sense of equation (276). If all of the delays are of size $1/4f_1$ only the fundamental is very effective in simulating the input signal; the amounts of the higher order harmonics in (276) provide very small envelope corrections to the waveform synthesized by b_0 and b_1 . This suggests the network shown in Fig. 19; the adaption procedure consists of first doing a first-order narrow-band adaption as previously described to obtain b_0 and b_1 for the first delay of $1/4f_1$ with b_2, \dots, b_M fixed at zero and the desired adaption signal being a monochromatic signal of frequency f_1 . When the gradient of the error function $\epsilon(k)$ decreases to a pre-assigned threshold value, the additional signal components are added to the desired adaption signal $s(t)$ and the weights $b_0, b_1, b_2, \dots, b_M$ are all allowed to adapt to steady state value. If the desired signal $s(t)$ is of the form shown in Fig. 18 only a very few additional weights b_2, \dots, b_M should be required and the adaption time should be very rapid.

4. The Use of Feedback Networks: The Kalman Filter

The work of Mantey^[41] indicates that it may be possible to produce a better adaptive synthesis procedure using feedback networks. However, he shows that the sum-squared quadratic error function

$$\sum_{k=0}^{\infty} [\epsilon_i(k)]^2 = \sum_{k=0}^{\infty} [s_i(k) - y_i(k)]^2 \quad (284)$$

possesses a unique minimum with respect to adaption only if the feedback coefficients a_1, \dots, a_N are all fixed. He then shows that modified adaption procedure in which the desired signal is used directly as the adaption signal applied to the feedback coefficients (Fig. 20), leads to a quadratic performance criterion which has a unique minimum. This new adaption procedure, in the case that $x(k)$ is white, has also been shown to be a stable network. [41,22]

Thus it would appear that perhaps substantial improvements in adaption performance might be realized by use of time variable feedback networks employing one or more feedback loops. The Kalman filter [46,47] is such a filter; it is a least-square-estimation filter which provides optimal estimates of the state vector $\underline{s}(k)$ recursively. The problem may be formulated here as that of obtaining the best estimate $\hat{\underline{s}}(k/k)$, of the state vector $\underline{s}(k)$ at time state k given k previous observations $x(k)$ and the previous optimal estimate $\hat{\underline{s}}(k/k-1)$ of the state vector. It is assumed that $\underline{s}(k)$ satisfies the dynamic system relations

$$\underline{s}(k) = \Phi(k, k-1) \underline{s}(k-1) + \Delta(k, k-1) \underline{n}(k) \quad (285)$$

$$\underline{x}(k) = H(k) \underline{s}(k) + \underline{y}(k) \quad (286)$$

where:

$\underline{s}(k)$ is the state vector (unknown) at time state k

$\Phi(k, k-1)$ is the state transition matrix (known)

$\Delta(k, k-1)$ is a known transition matrix

$H(k)$ is a known transformation matrix

$\underline{x}(k)$ is the observation vector

$\underline{n}(k), \underline{y}(k)$ are sequences of independent Gaussian random vectors with zero means and the known covariance matrices

$$E[\underline{n}(k) \underline{n}^*(\ell)] = Q \delta_{k,\ell} \quad E[\underline{v}(k) \underline{v}^*(\ell)] = R \delta_{k,\ell}$$

$$E[\underline{n}(k) \underline{v}^*(\ell)] = 0. \quad (287)$$

They represent assumed additive excitation and measurement noise signals respectively.

Following Lee,^[47] the optimal estimate $\hat{\underline{s}}(k/k)$ is assumed to be the sum of the extrapolated estimate, based on past estimates and a weighting "gain" $G(k)$, times the residual difference between the new measurement and the extrapolated previous measurement,

$$\hat{\underline{s}}(k/k) = \Phi \hat{\underline{s}}(k/k-1) + G(k) [\underline{x}(k) - H \hat{\underline{s}}(k/k-1)]. \quad (288)$$

This can be re-written in the form

$$\hat{\underline{s}}(k/k) = [\Phi - GH\Phi] \hat{\underline{s}}(k-1/k-1) + G \underline{x}(k). \quad (289)$$

This suggests in the case of two equal time delays the state transition flow diagram shown in Fig. 21. The linear equations equivalent to (289) are, from Fig. 21,

$$\hat{s}_1(k/k) = [\phi_{11} - g_{11}\phi_{11}] \hat{s}_1(k-1/k-1) + [\phi_{12} - g_{11}\phi_{12}] \hat{s}_2(k-1/k-1) + g_{11}x(k) \quad (290)$$

$$\hat{s}_2(k/k) = [\phi_{21} - g_{21}\phi_{11}] \hat{s}_1(k-1/k-1) + [\phi_{22} - g_{21}\phi_{12}] \hat{s}_2(k-1/k-1) + g_{21}x(k) \quad (291)$$

$$\hat{s}_1(k-1/k-1) = z^{-1} \hat{s}_1(k/k) \quad (292)$$

$$\hat{s}_2(k-1/k-1) = z^{-1} \hat{s}_2(k/k) \quad (293)$$

where

$$G = \begin{pmatrix} g_{11} & g_{12} \\ g_{21} & g_{22} \end{pmatrix}, \quad H = \begin{pmatrix} 1 & 0 \\ 0 & 0 \end{pmatrix}, \quad \Phi = \begin{pmatrix} \phi_{11} & \phi_{12} \\ \phi_{21} & \phi_{22} \end{pmatrix}. \quad (294)$$

Since the desired output is $\hat{s}_1(k/k)$, the system (290,293) can be solved for the transfer function $\hat{s}_1(k/k) / x(k)$,

$$\begin{aligned} \frac{\hat{s}_1(k/k)}{x(k)} &= \frac{g_{11}[1-(\phi_{22}-g_{21}\phi_{12})z^{-1}] + g_{21}[\phi_{12}]z^{-1}}{\Psi_1} \\ &= \frac{g_{11}[\phi_{22}-g_{21}\phi_{12}-\phi_{12}+g_{11}\phi_{12}]z^{-1}}{\Psi_2} \end{aligned} \quad (295)$$

where

$$\begin{aligned} \Psi_1 &= (1-(\phi_{11}-g_{11}\phi_{11})z^{-1})(1-(\phi_{22}-g_{21}\phi_{12})z^{-1}) - (\phi_{12}-g_{11}\phi_{12}) \\ &\quad \cdot (\phi_{21}-g_{21}\phi_{11})z^{-2} \end{aligned}$$

$$\begin{aligned} \Psi_2 &= 1 - [\phi_{11}-g_{11}\phi_{11} + \phi_{22}-g_{21}\phi_{12}]z^{-1} + [(\phi_{11}-g_{11}\phi_{11})(\phi_{22}-g_{21}\phi_{12}) \\ &\quad - (\phi_{12}-g_{11}\phi_{12})(\phi_{21}-g_{21}\phi_{11})]z^{-2} . \end{aligned}$$

This, by comparison with (274) is equivalent to the sample data system shown in Fig. 22. For any desired signal $s(k)$ it is possible to thus obtain the optimal least-squares estimate of $\hat{s}_1(k/k)$ at the output of the network provided

$$b_0(k) = g_{11}(k) \quad (298)$$

$$b_1(k) = [\phi_{22}(k)-g_{21}(k)\phi_{12}(k)-\phi_{12}(k)+g_{21}(k)\phi_{12}(k)] \quad (299)$$

$$a_1(k) = [\phi_{11}(k)-g_{11}(k)\phi_{11}(k)+\phi_{22}(k)-g_{21}(k)\phi_{12}(k)] \quad (300)$$

$$\begin{aligned} a_2(k) &= [(\phi_{11}(k)-g_{11}(k)\phi_{11}(k))(\phi_{22}(k)-g_{21}(k)\phi_{12}(k)) \\ &\quad - (\phi_{12}(k)-g_{11}(k)\phi_{12}(k))(\phi_{21}(k)-g_{21}(k)\phi_{11}(k))] . \end{aligned} \quad (301)$$

If the signal process $s(k)$ is stationary, the matrix Φ will be

time invariant and the gain matrix $G(k)$ will quite soon reach steady states; this leads to the exciting possibility that the steady state pre-calculated values for a_1, a_2, b_o, b_1 may be used directly in (295). Although this would not give optimal estimates during the usual transient phase of the adaption procedure, previous experience with the technique of using fixed gains indicates that it begins providing optimal estimates after a very short time.

VII. CONCLUSIONS

This research has concerned investigations into the inter-related topics of large-domain integration techniques for solution to the scalar multi-dimensional variable-parameter acoustic wave equation and adaptive antenna-array processors and algorithms.

After formulation of the acoustic propagation medium equations to be integrated in properly posed form, their solution by means of various finite-difference integration techniques was investigated. Among the results that are considered new or significant are the following.

A method for "tearing"^[45] or partitioning very large variable parameter domains in cases where steady state propagation characteristic are desired is given. Such positioning is essential if the integration of the wave equation for large domains is to be carried out on reasonable-sized machines or in background mode in a multi-tasking environment. Computer calculations verify that consistent results are obtainable using the method, but its practical use depends upon the efficiency of the integration scheme chosen and "asking the right questions." Since, inherently, the method is capable of giving the complete steady-state pressure velocity distribution profiles at all points of space it is not surprising that quite large computing time requirements result. However, if the integration is confined by program control to only those regions of the spacial domain wherein the state variables are above an assigned threshold, great reductions in computing time are possible. Indeed there appears no a priori reason why ray-path analysis cannot be used for regions wherein the properties are known to be quite regular with a switch to direct

wave-equation integration when approaching regions of unpredictable behavior such as the incidence of convergence zones, highly variable parameter domains, defraction regions, or where three-dimensional behavior cannot be ignored.

A new method for integrating the wave-equation in three dimensions is developed which uses the Fast Fourier Transform digital algorithm. The method is fast, accurate, and easily implemented for general rectangular domains but as with all Fourier techniques is restricted to constant parameter regions. The method is semi-implicit and appears to possess considerable stability even for fairly large integration step sizes.

The Douglas and Gunn tri-diagonal integration scheme was implemented for the wave equation in three dimensions. The scheme is faster computationally than any of the other algorithms tried in the case of variable coefficient problems. However, it requires a great deal of programming effort, large amounts of work storage, and the conditions of stability are not precisely known.

As an aid in understanding and applying the general discrete Fourier Transform, a new method for finding analytic closed form expressions for the coefficients is given. The method is basically an extension to discrete domains of the impulse function method of continuous distribution theory. Although the method does not guarantee that the investigators efforts will be small, it appears that coefficient expressions can be obtained quite easily for functions having relatively few discontinuities in analytic-envelope.

Relative to processors which seem especially appropriate for use in variable parameter environments the topic of adaptive array processors and algorithms was considered. First, the classical optimum statistical estimation procedure as it applies to array processors was reviewed. In the classical analog approach it is shown that discrete spectrum algorithms of the FFT type can be substituted provided in the analogous continuous spectrum variables proper precautions are observed relative to estimation and/or detection interval T and sampling interval Δt . This generally will require that N , the number of sampling points for detection interval T , be quite large; this is no particular restriction and in fact the FFT algorithm approaches maximum efficiency for large N .

The second point of view is that leading to automatic adaptive synthesis procedures in which quadratic-error functional minimizations are carried out. The Widrow scheme is considered as a special case of a general feed-forward, feedback network of the type considered extensively by Mantey. Results of a visual display simulation program are reported and the use of other possible adaptive algorithms such as variable delays and the Kalman filter are discussed.

APPENDIX A

THE DOUGLAS AND GUNN EQUATIONS

The Douglas and Gunn equations iteratively solve the wave equation in three dimensions by computing a series of three approximations to the state components at the time $(r+1) \Delta t$ in terms of state components at the time $r \Delta t$. The particular equations to be solved for any one approximation are tri-diagonal in terms of the implicit variables; the solution to the implicit portion of the previous approximation acts as an explicit input to the next approximation. Thus the implicit system of equations to be solved at any stage are tri-diagonal and hence, in the case of the three dimensional wave equation the work of solving the system is only slightly greater than three times the work required to solve the equivalent one-dimensional wave equation.

Denoting by $\underline{u}(\ell, m, n, r)$ the known state of the system at position $\ell \Delta x, m \Delta y, n \Delta z, r \Delta t$, by $\underline{u}^*(\ell, m, n, r+1)$ the first approximation to the solution of the wave equation at state $(r+1) \Delta t$, by $\underline{u}^{**}(\ell, m, n, r+1)$ the second approximation, etc., the equations to be solved are shown below. For purposes of brevity only the indices that vary in any given term are explicitly written down.

$$p^*(r+1)-p = -\alpha [v_x^*(\ell+1, r+1) - v_x^*(\ell-1, r+1) + v_x(\ell+1) - v_x(\ell-1) + 2(v_y(m+1) - v_y(m-1) + v_z(m+1) - v_z(n-1))] \quad (A1)$$

$$v_x^*(r+1) - v_x = -\alpha [p^*(\ell+1, r+1) - p^*(\ell-1, r+1) + p(\ell+1) - p(\ell-1)] \quad (A2)$$

$$v_y^*(r+1) - v_y = -2\alpha [p(m+1) - p(m-1)] \quad (A3)$$

$$v_z^*(r+1) - v_z = -2\alpha [p(n+1) - p(n-1)] \quad (A4)$$

End of first approximation equations

$$p^{**}(r+1) - p = -\alpha [v_x^*(\ell+1, r+1) - v_x^*(\ell-1, r+1) + v_x(\ell+1) - v_x(\ell-1) + v_y^{**}(m+1, r+1) - v_y^{**}(m-1, r+1) + v_y(m+1) - v_y(m-1) + 2(v_z(n+1) - v_z(n-1))] \quad (A5)$$

$$v_x^{**}(r+1) - v_x = -\alpha [p^*(\ell+1, r+1) - p^*(\ell-1, r+1) + p(\ell+1) - p(\ell-1)] \quad (A6)$$

$$v_y^{**}(r+1) - v_y = -\alpha [p^{**}(m+1, r+1) - p^{**}(m-1, r+1) + p(\ell+1) - p(\ell-1)] \quad (A7)$$

$$v_z^{**}(r+1) - v_z = -2\alpha [p(n+1) - p(n-1)] \quad (A8)$$

End of second approximation equations

$$p(r+1) - p = -\alpha [v_x^*(\ell+1, r+1) - v_x^*(\ell-1, r+1) + v_x(\ell+1) - v_x(\ell-1) + v_y^{**}(m+1, r+1) - v_y^{**}(m-1, r+1) + v_y(m+1) - v_y(m-1) + v_z(m+1, r+1) - v_z(n-1, r+1) + v_z(n+1) - v_z(n-1)] \quad (A9)$$

$$v_x(r+1) - v_x = -\alpha [p^*(\ell+1, r+1) - p^*(\ell-1, r+1) + p(\ell+1) - p(\ell-1)] \quad (A10)$$

$$v_y(r+1) - v_y = -\alpha [p^{**}(m+1, r+1) - p^{**}(m-1, r+1) + p(m+1) - p(m-1)] \quad (A11)$$

$$v_z(r+1) - v_z = -\alpha [p(n+1, r+1) - p(n-1, r+1) + p(n+1) - p(n-1)] \quad (A12)$$

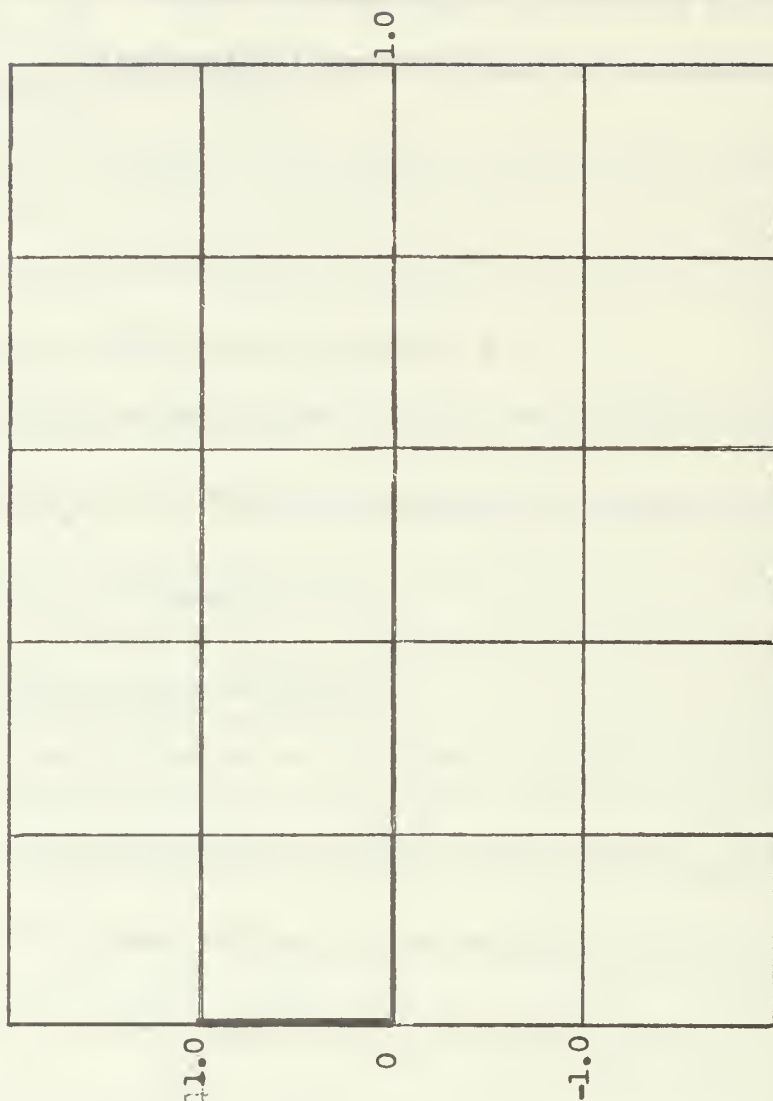
where

$$\alpha = - \frac{c \Delta t}{4 \Delta x} \quad (A13)$$

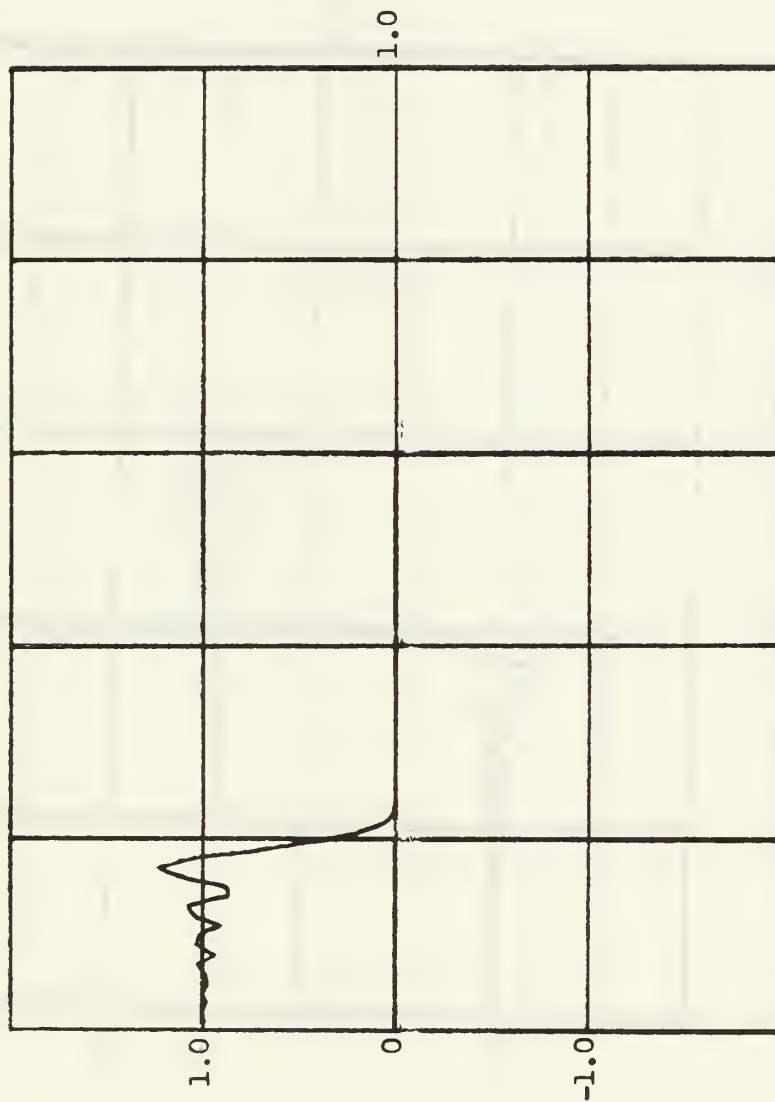
APPENDIX B

THE SIGNAL INPUT TO A MATCHED LINE

This appendix shows the response of a matched output one-dimensional domain to a step input function applied at time $t=0$ sec. The domain is of unit length and is subdivided into 100 segments.



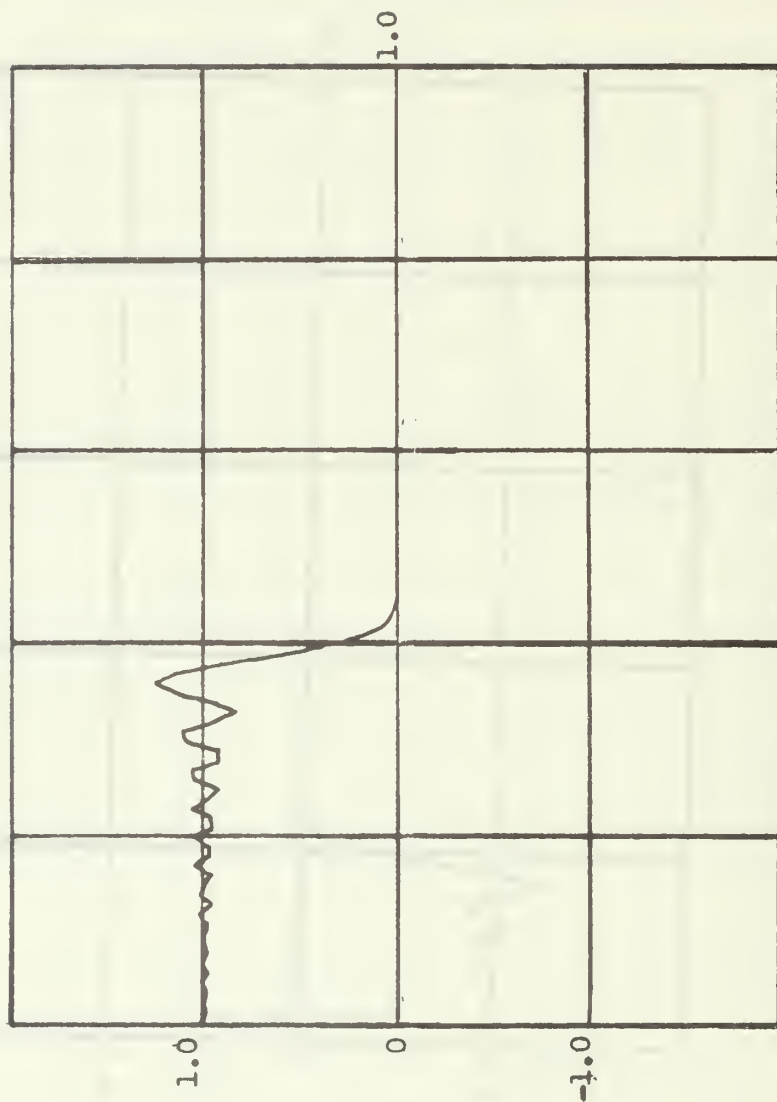
Horizontal scale = 2.00E-01 units/inch
 Vertical scale = 1.00E+00 units/inch
 Spacial pressure distribution at $t = 0$ sec. of a matched
 1-dimensional domain to a unit step input. Length=1.0 unit. $dx=0.01$



Horizontal scale = $2.00\text{E}-01$ units/inch

Vertical scale = $1.00\text{E}+00$ units/inch

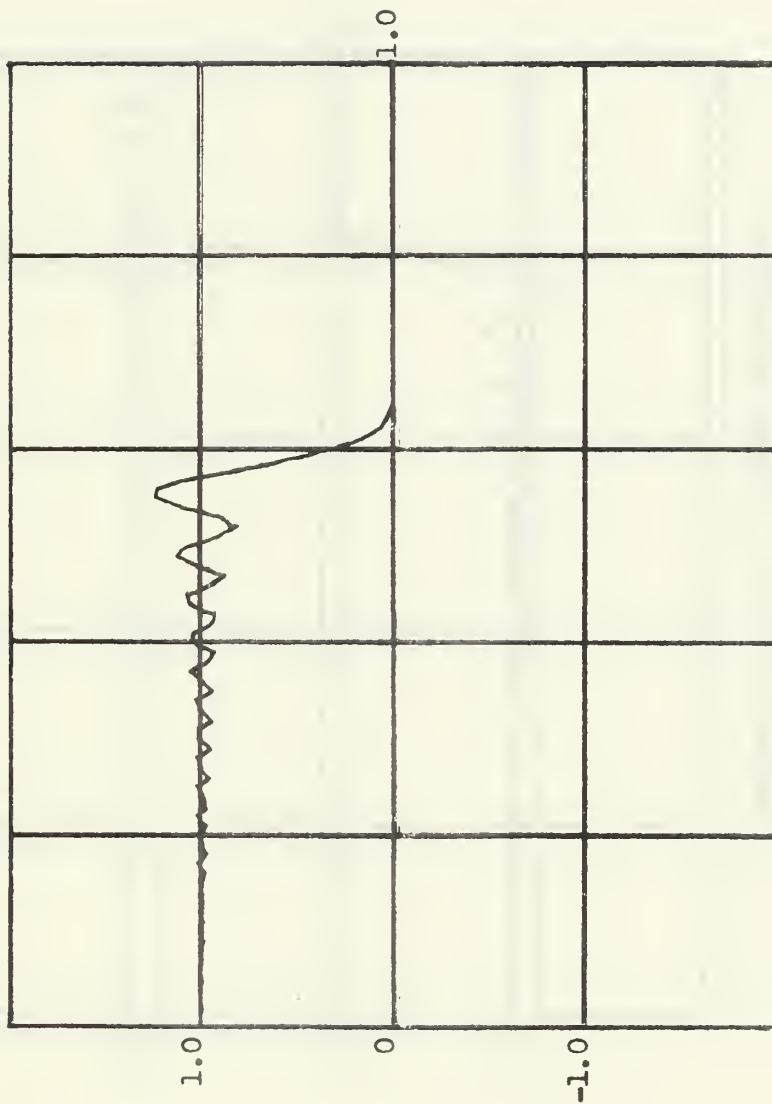
Spatial pressure distribution at $t = 20$ sec. of a matched
1-dimensional domain to a unit step input. Length=1.0 unit. $dx=0.01$



Horizontal scale = $2.00E-01$ units/inch

Vertical scale = $1.00E+00$ units/inch

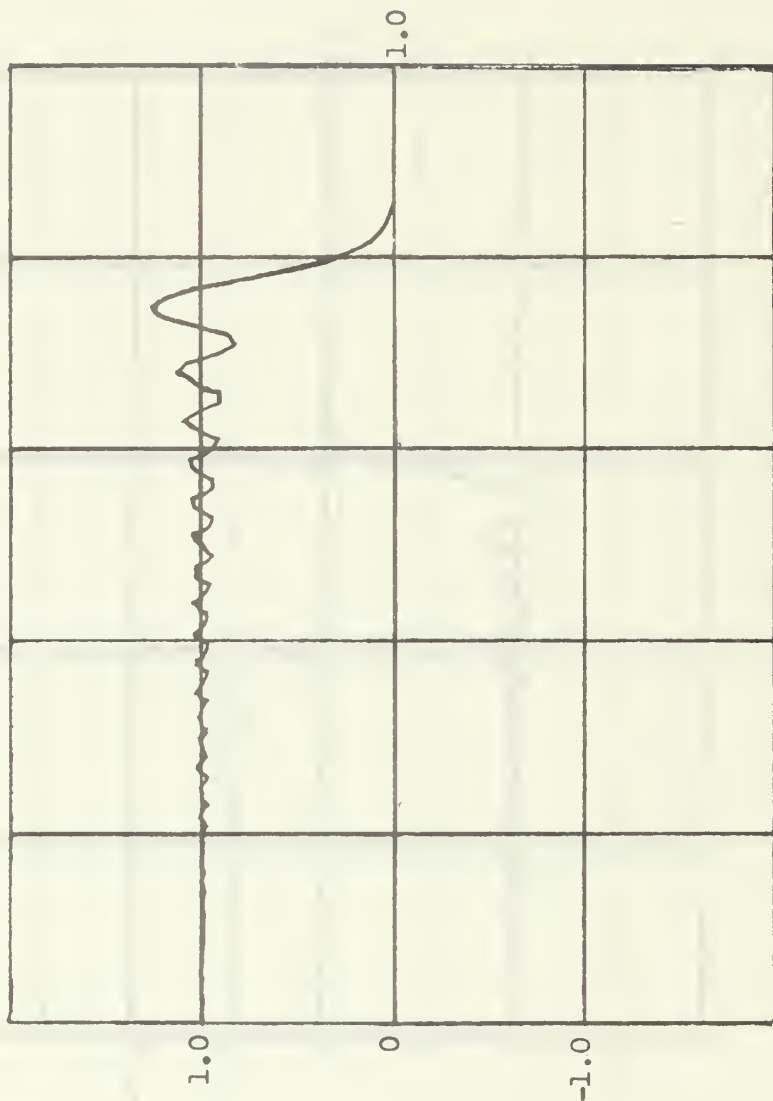
Spatial pressure distribution at $t = .40$ sec. of a matched 1-dimensional domain to a unit step input. Length=1.0 unit. $dx=0.01$



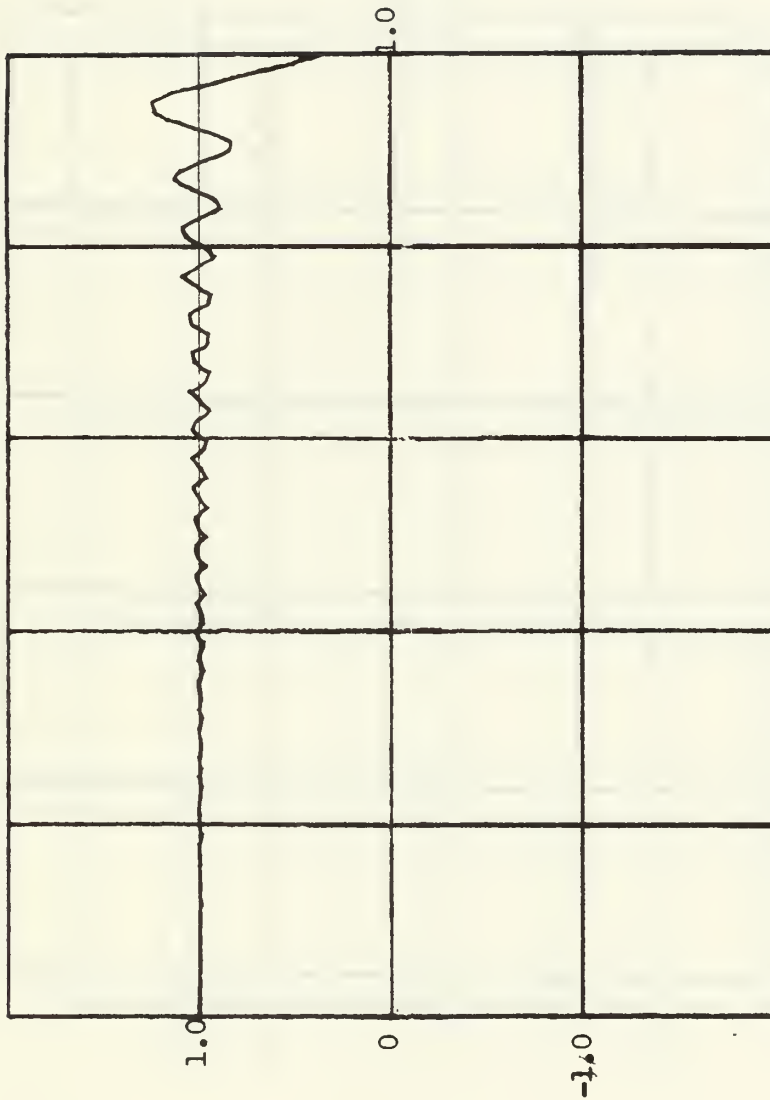
Horizontal scale = $2.00E-01$ units/inch

Vertical scale = $1.00E+00$ units/inch

Spatial pressure distribution at $t = .60$ sec. of a matched 1-dimensional domain to a unit step input. Length=1.0 unit. $dx=0.01$



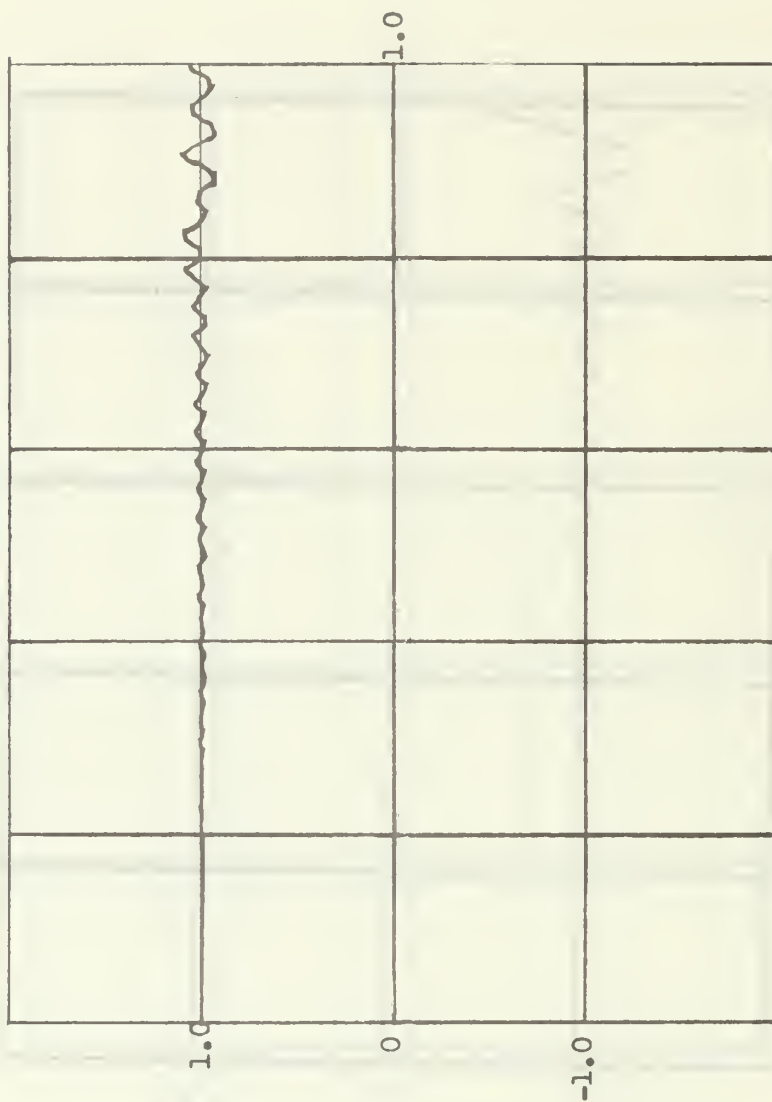
Horizontal scale = $2.00E-01$ units/inch
 Vertical scale = $1.00E+00$ units/inch
 Spatial pressure distribution at $t = .80$ sec. of a matched
 1-dimensional domain to a unit step input. Length=1.0 unit. $dx=0.01$



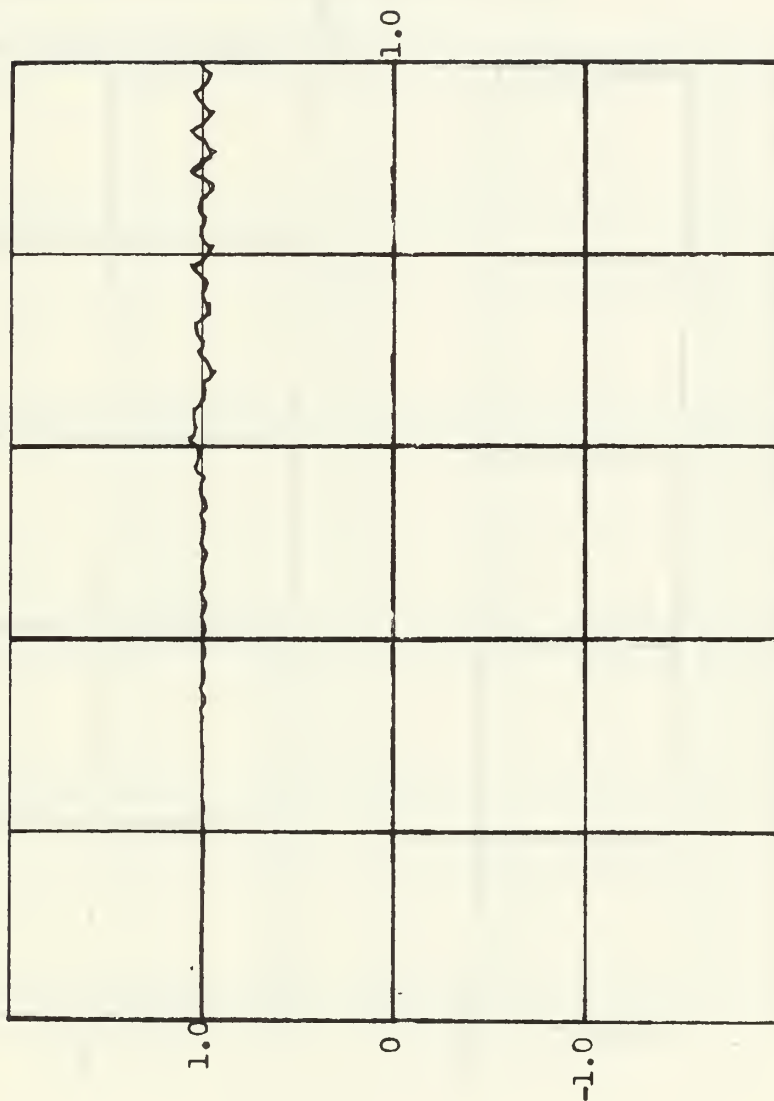
Horizontal scale = 2.00E-01 units/inch

Vertical scale = 1.00E+00 units/inch

Spatial pressure distribution at $t = 1.0$ sec. of a matched 1-dimensional domain to a unit step input. Length=1.0 unit. $dx=0.01$



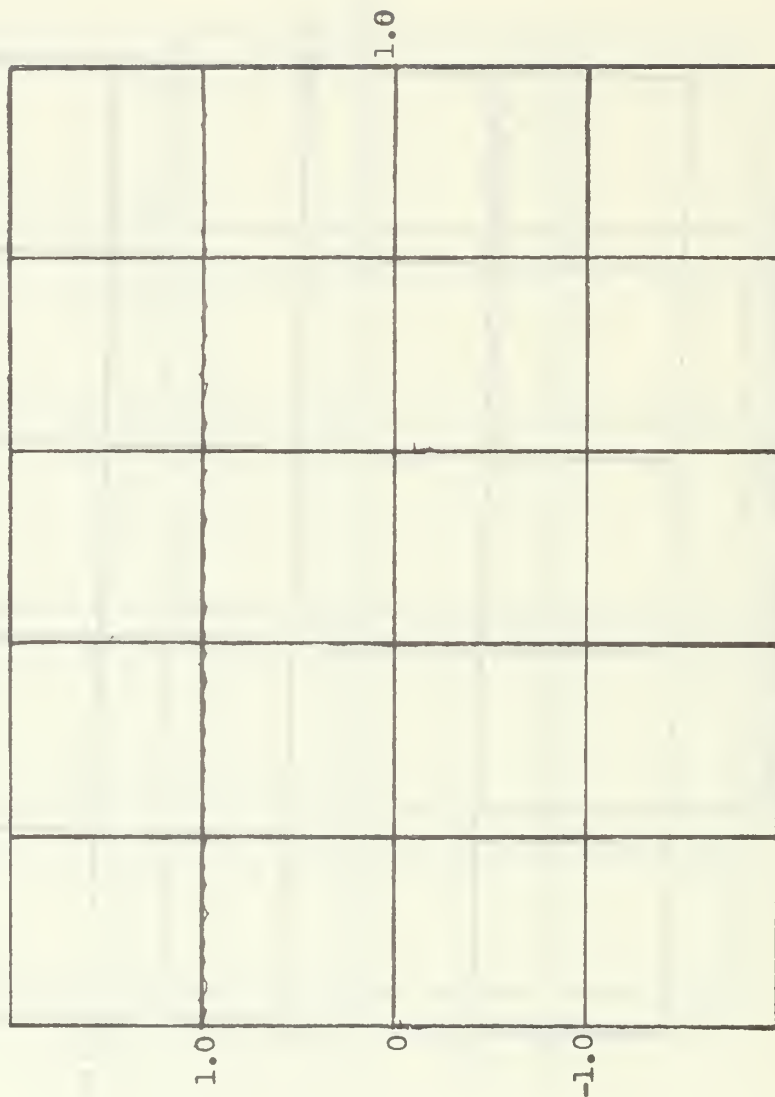
Horizontal scale = $2.00E-01$ units/inch
 Vertical scale = $1.00E+00$ units/inch
 Spatial pressure distribution at $t = 1.20$ sec. of a matched
 1-dimensional domain to a unit step input. Length=1.0 unit. $dx=0.01$



Horizontal scale = $2.00\text{E-}01$ units/inch

Vertical scale = $1.00\text{E-}01$ units/inch

Spacial pressure distribution at $t = 1.40$ sec. of a matched
1-dimensional domain to a unit step input. Length=1.0 unit. $dx=0.01$



Horizontal scale = 2.00E-01 units/inch

Vertical scale = 1.00E+00 units/inch

Special pressure distribution at $t = 6.0$ sec. of a matched 1-dimensional domain to a unit step input. Length=1.0 unit. $dx=0.01$

APPENDIX C

THE COSINE INPUT TO A MATCHED LINE

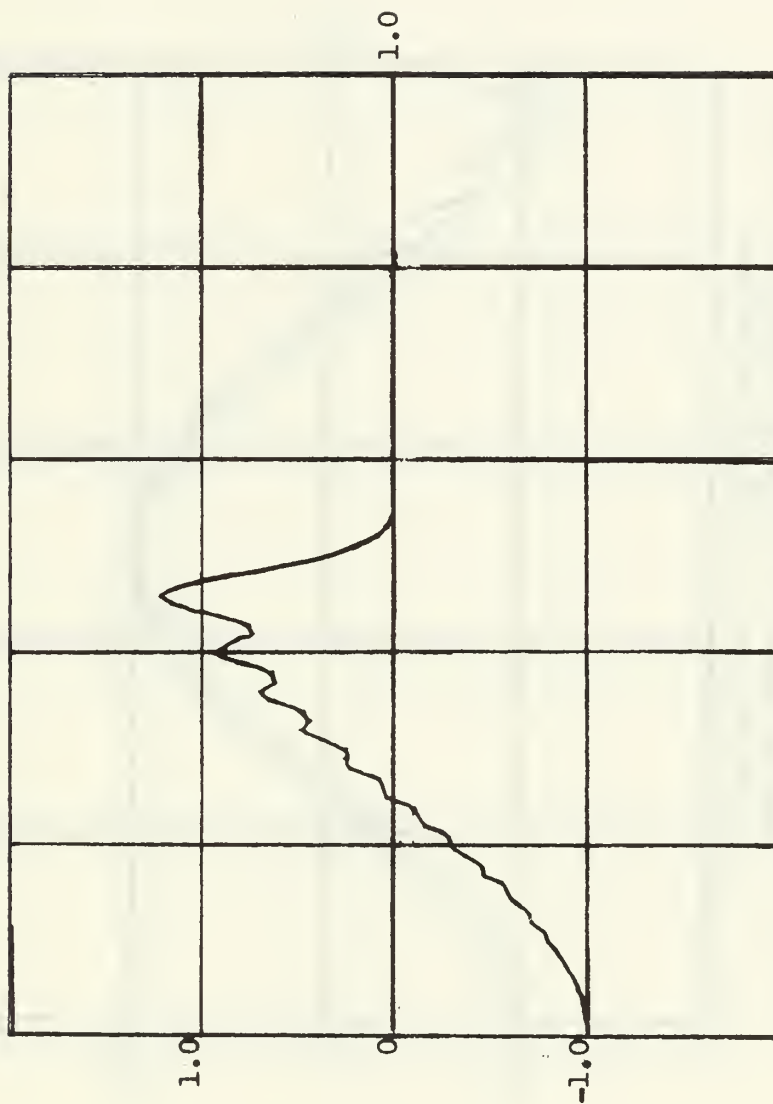
This appendix shows the response to the same domain as described in Appendix B but with a 1 Hz cosine signal input.



Horizontal scale = 2.00E-01 units/inch

Vertical scale = 1.00E+00 units/inch

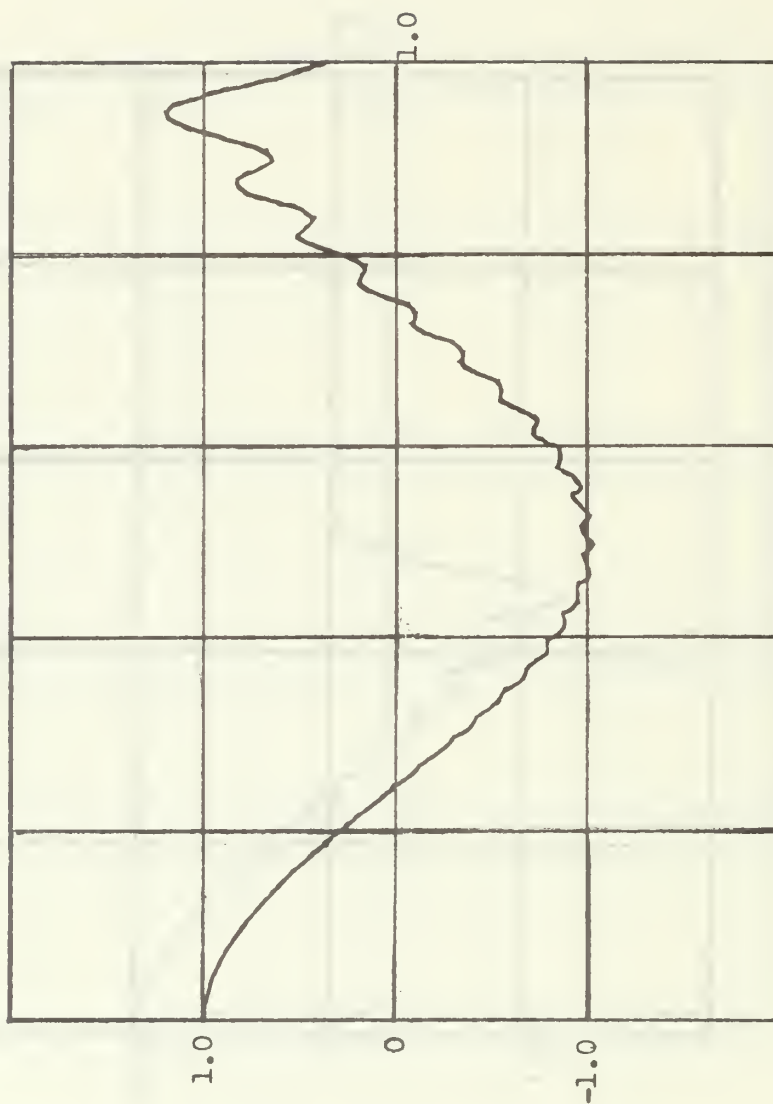
Spacial pressure distribution at $t = 0.00$ sec. of a matched 1-dimensional domain to a unit step input. Length=1.0 unit. $dx=0.01$



Horizontal scale = $2.00E-01$ units/inch

Vertical scale = $1.00E+00$ units/inch

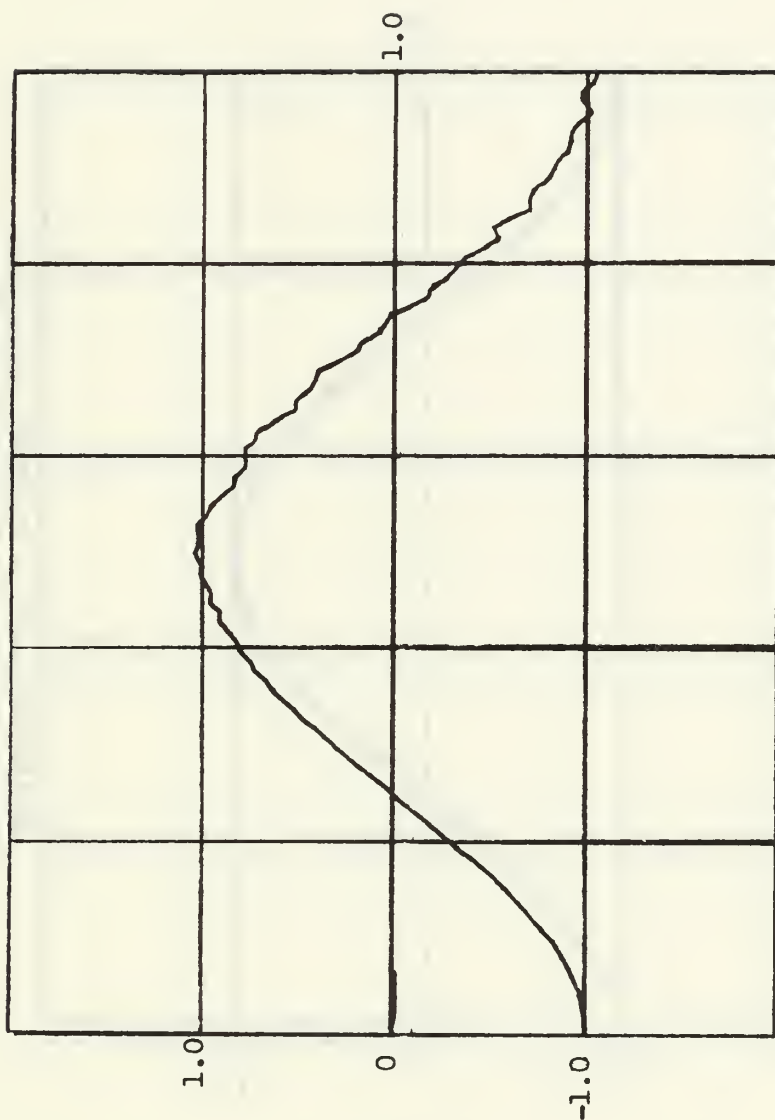
Spatial pressure distribution at $t = .50$ sec. of a matched
1-dimensional domain to a unit step input. Length=1.0 unit. $dx=0.01$



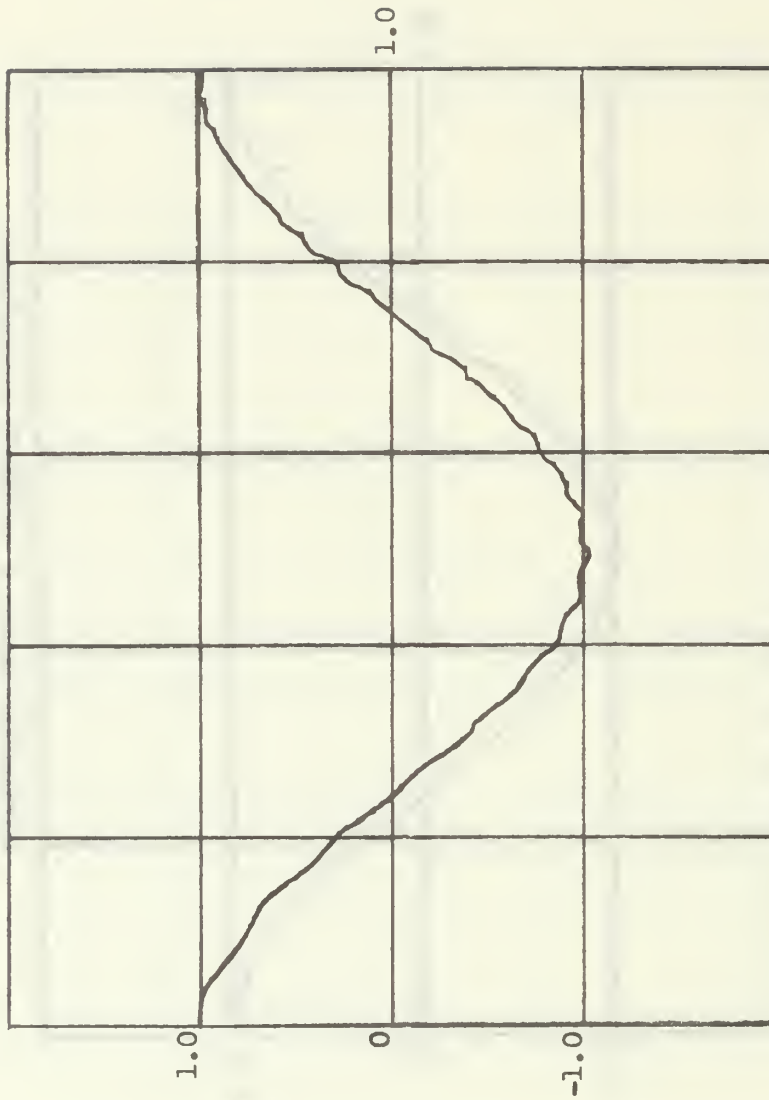
Horizontal scale = 2.00E-01 units/inch

Vertical scale = 1.00E+00 units/inch

Spatial pressure distribution at $t = 1.00$ sec. of a matched 1-dimensional domain to a unit step input. Length=1.0 unit. $dx=0.01$



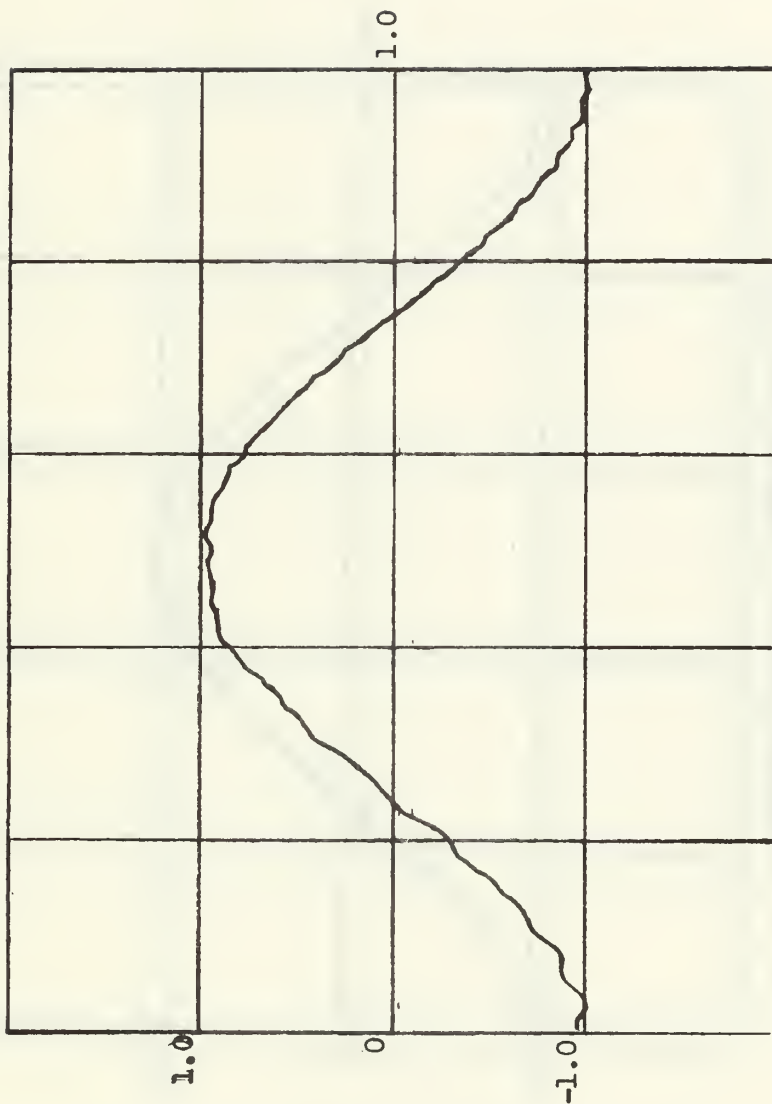
Horizontal scale = $2.00\text{E}-01$ units/inch
 Vertical scale = $1.00\text{E}+00$ units/inch
 Spatial pressure distribution at $t = 1.50$ sec. of a matched
 1-dimensional domain to a unit step input. Length=1.0 unit. $dx=0.01$



Horizontal scale = $2.00\text{E}-01$ units/inch

Vertical scale = $1.00\text{E}+00$ units/inch

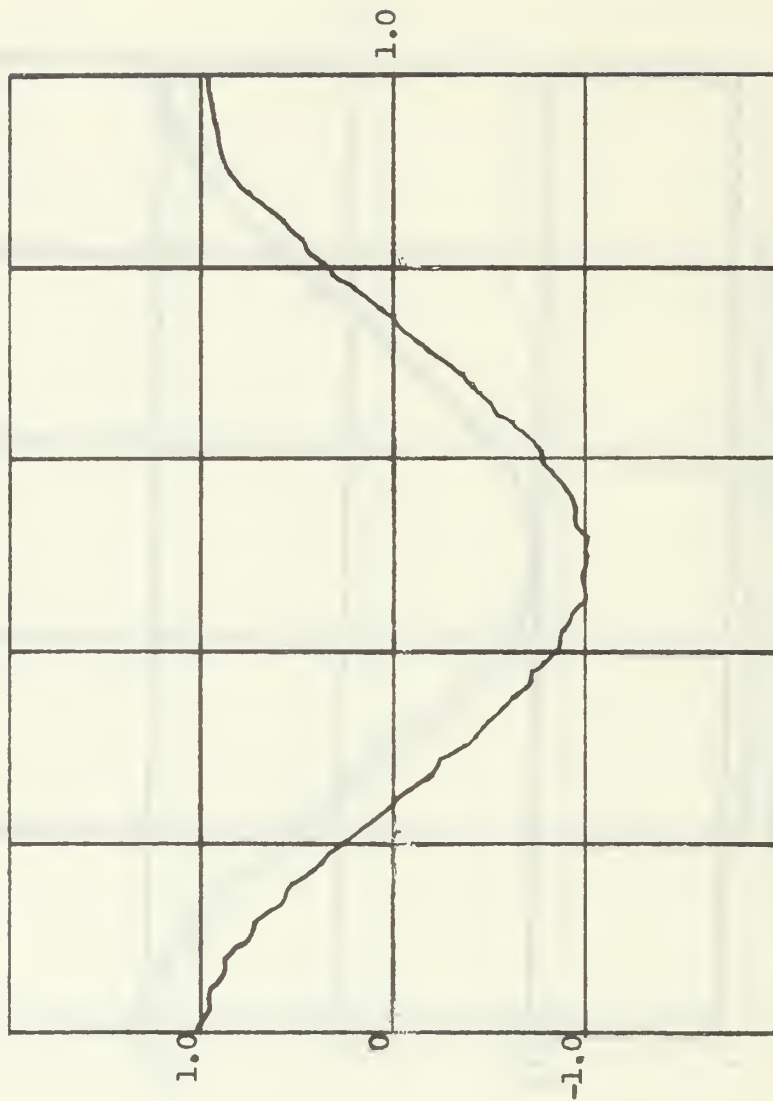
Spatial pressure distribution at $t = 2.00$ sec. of a matched 1-dimensional domain to a unit step input. Length=1.0 unit. $dx=0.01$



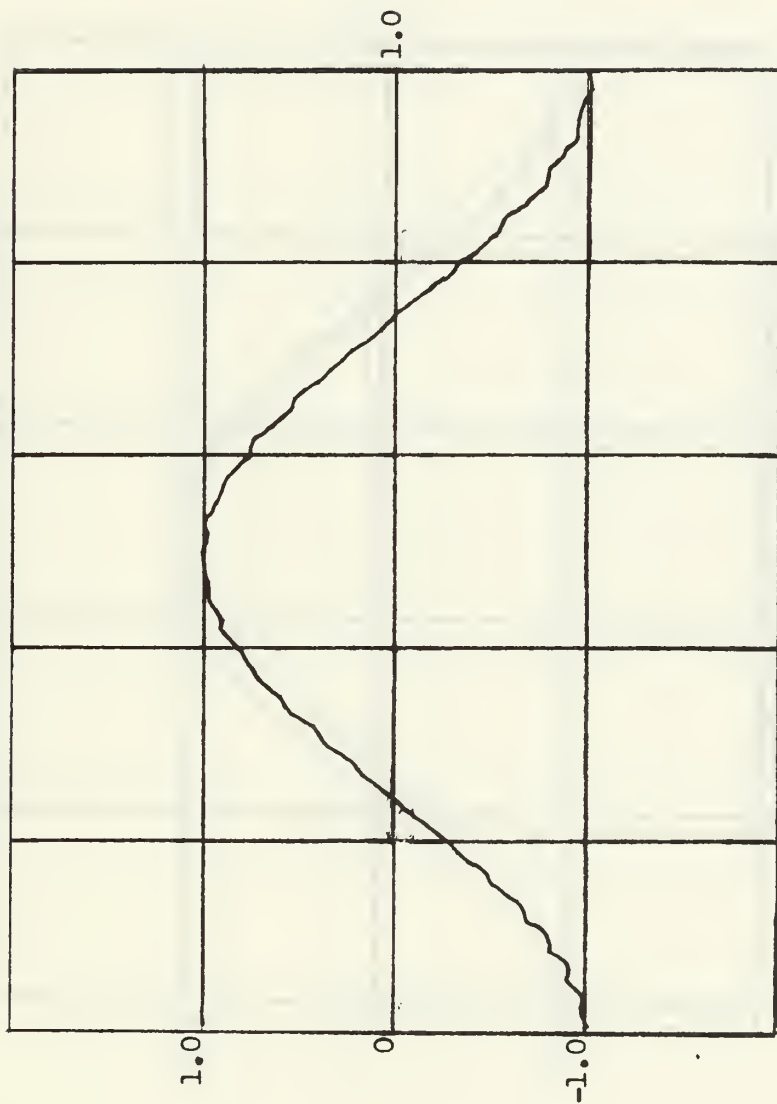
Horizontal scale = 2.00E-01 units/inch

Vertical scale = 1.00E+00 units/inch

Spatial pressure distribution at $t = 2.50$ sec. of a matched 1-dimensional domain to a unit step input. Length=1.0 unit. $dx=0.01$



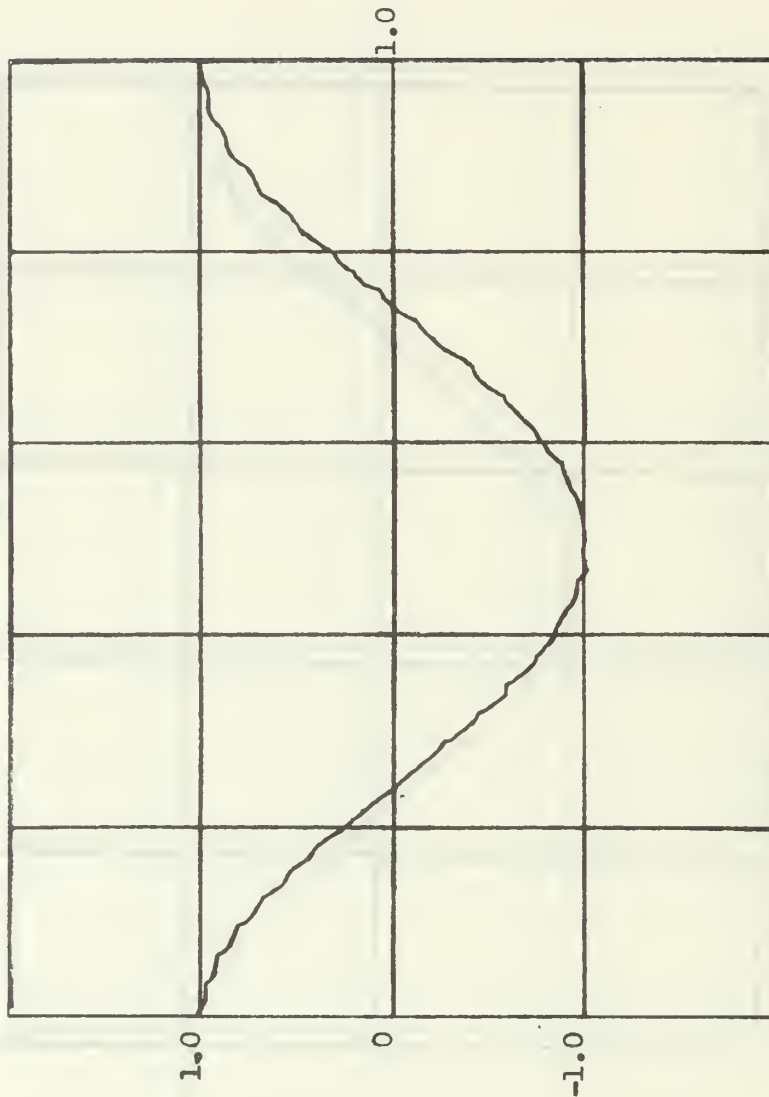
Horizontal scale = 2.00E-01 units/inch
Vertical scale = 1.00E+00 units/inch
Spatial pressure distribution at $t = 3.00$ sec. of a matched
1-dimensional domain to a unit step input. Length=1.0 unit. $dx=0.01$



Horizontal scale = 2.00E-01 units/inch

Vertical scale = 1.00E+00 units/inch

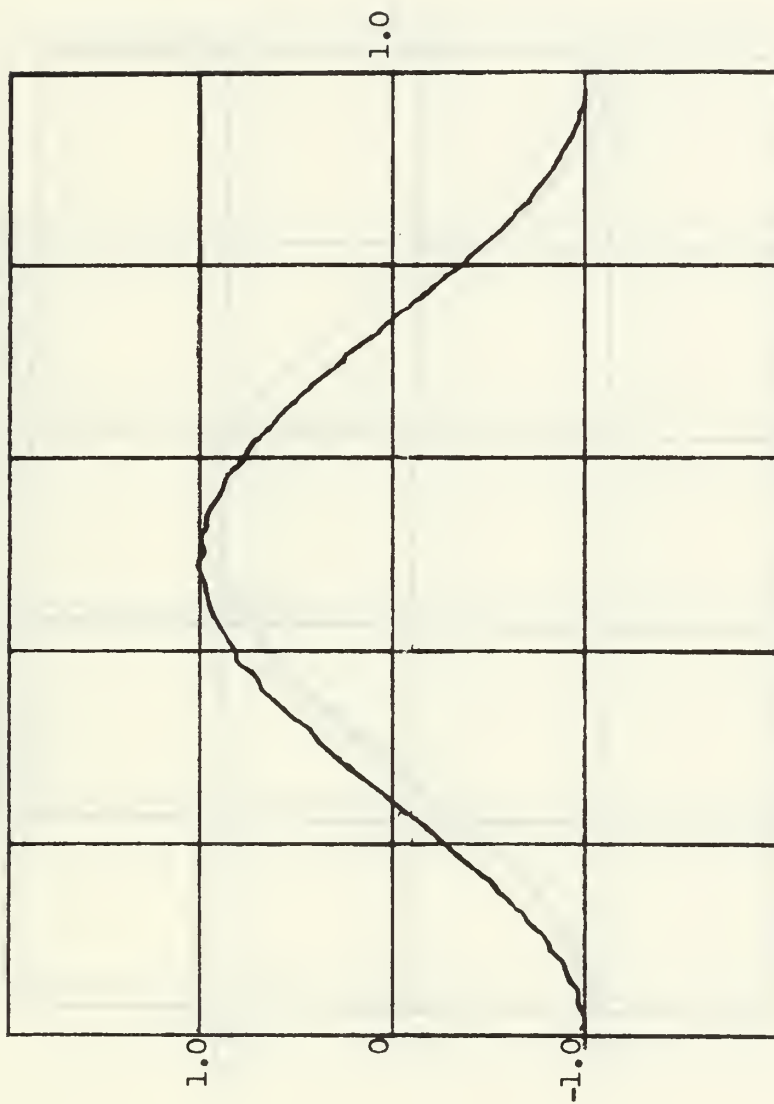
Spacial pressure distribution at $t = 3.50$ sec. of a matched 1-dimensional domain to a unit step input. Length=1.0 unit. $dx=0.01$



Horizontal scale = 2.00E-01 units/inch

Vertical scale = 1.00E+00 units/inch

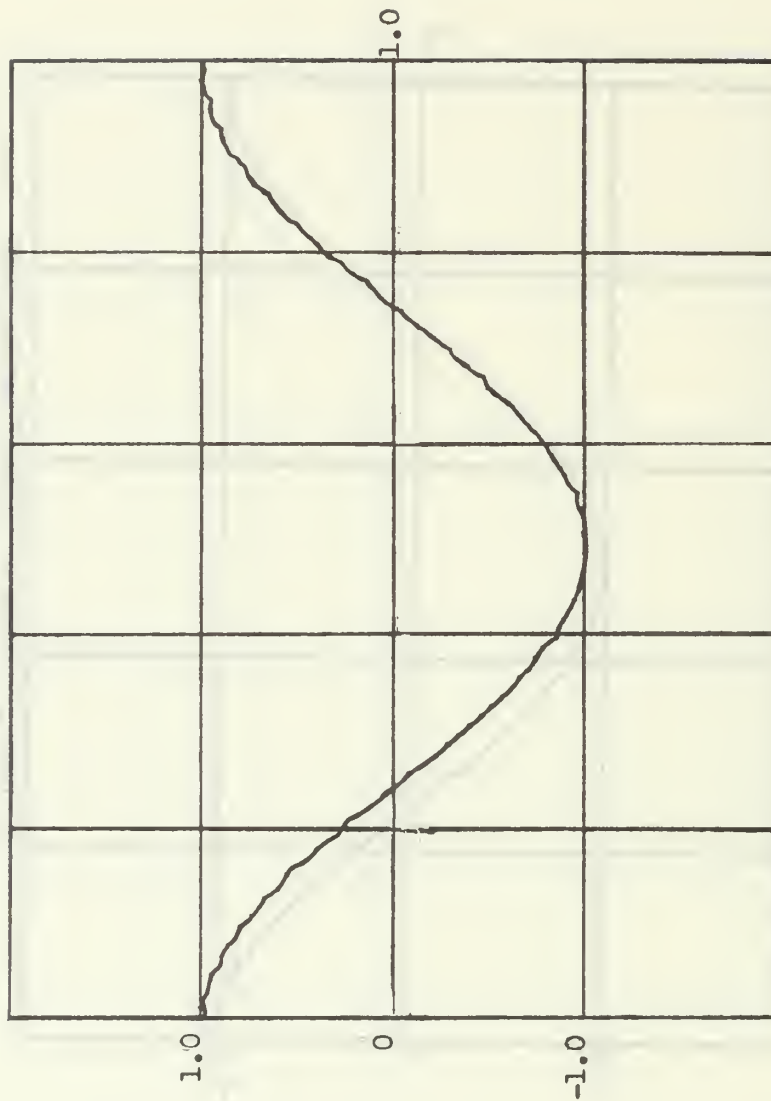
Spatial pressure distribution at $t = 4.00$ sec. of a matched 1-dimensional domain to a unit step input. Length=1.0 unit. $dx=0.01$



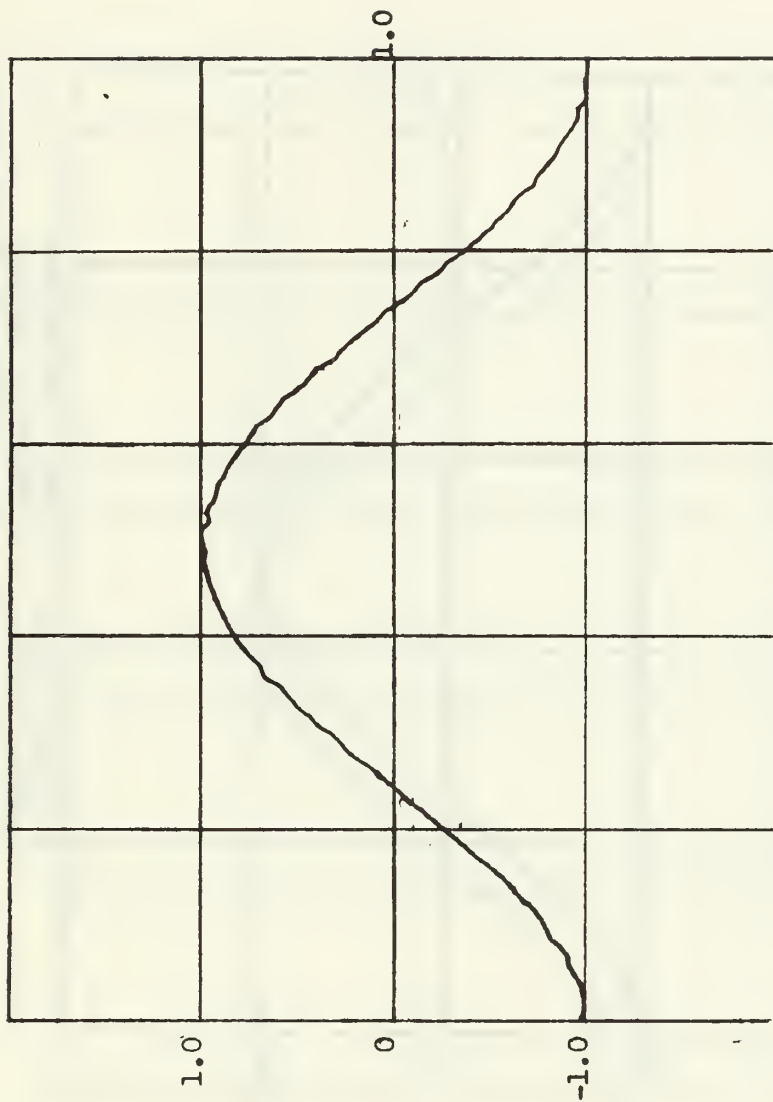
Horizontal scale = $2.00E-01$ units/inch

Vertical scale = $1.00E+00$ units/inch

Spatial pressure distribution at $t = 4.50$ sec. of a matched 1-dimensional domain to a unit step input. $dx=0.01$



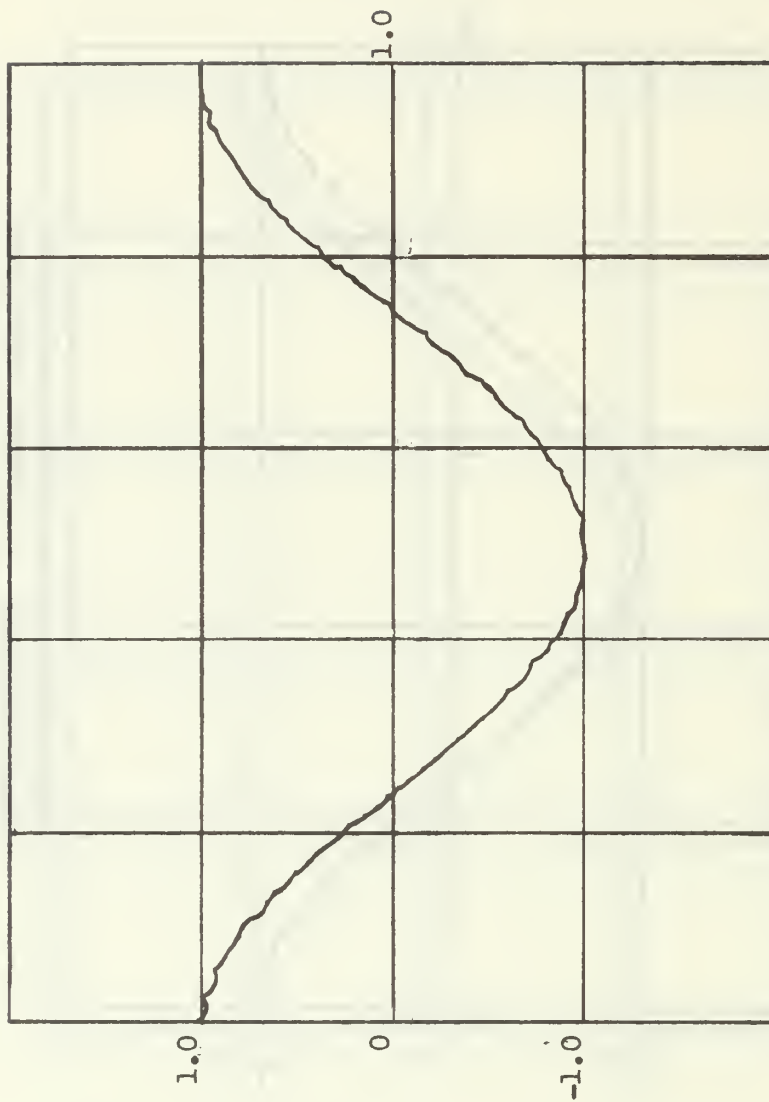
Horizontal scale = 2.00E-01 units/inch
 Vertical scale = 1.00E+00 units/inch
 Spatial pressure distribution at $t = 5.00$ sec. of a matched
 1-dimensional domain to a unit step input. Length=1.0 unit. $dx=0.01$



Horizontal scale = $2.00\text{E}-01$ units/inch

Vertical scale = $1.00\text{E}+00$ units/inch

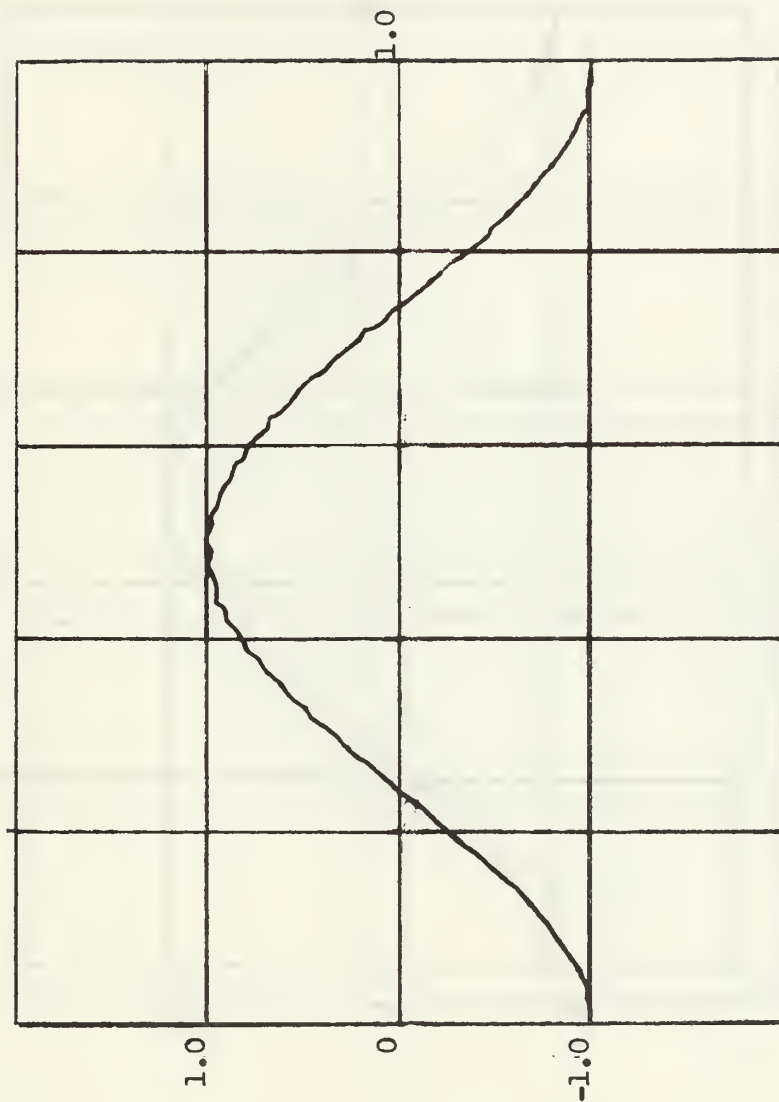
Spatial pressure distribution at $t = 5.50$ sec. of a matched 1-dimensional domain to a unit step input. $dx=0.01$



Horizontal scale = 2.00E-01 units/inch

Vertical scale = 1.00E+00 units/inch

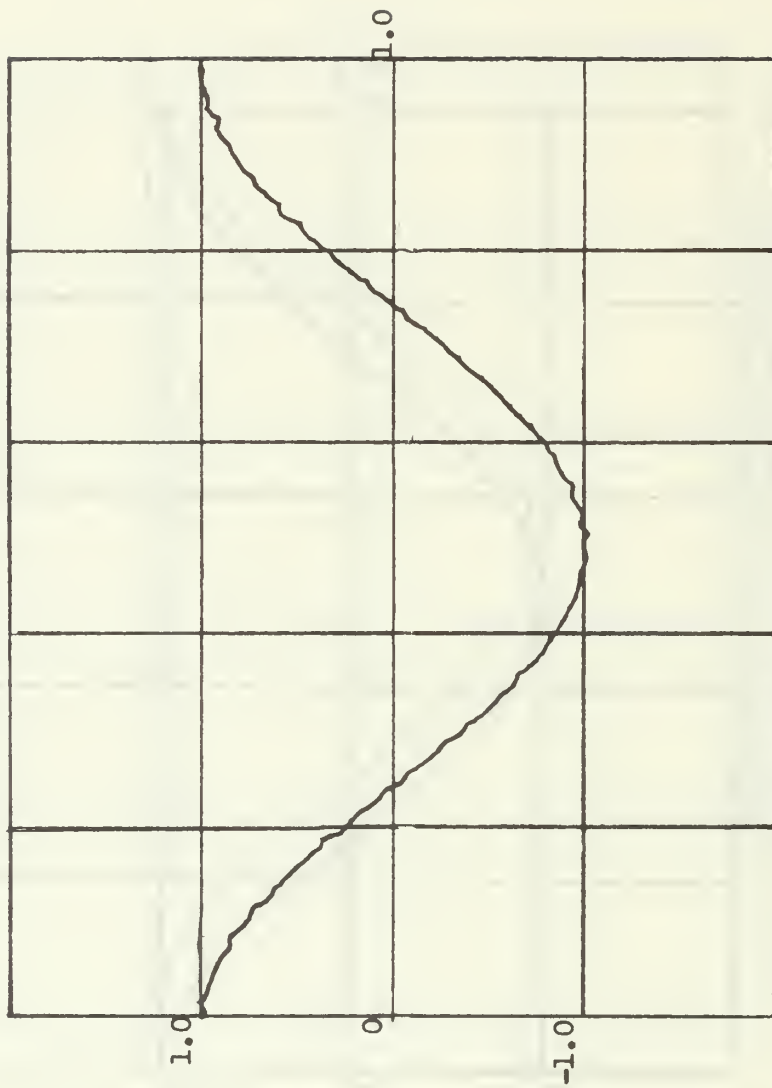
Spatial pressure distribution at $t = 6.00$ sec. of a matched 1-dimensional domain to a unit step input. Length=1.0 unit. $dx=0.01$



Horizontal scale = $2.00\text{E}-01$ units/inch

Vertical scale = $1.00\text{E}+00$ units/inch

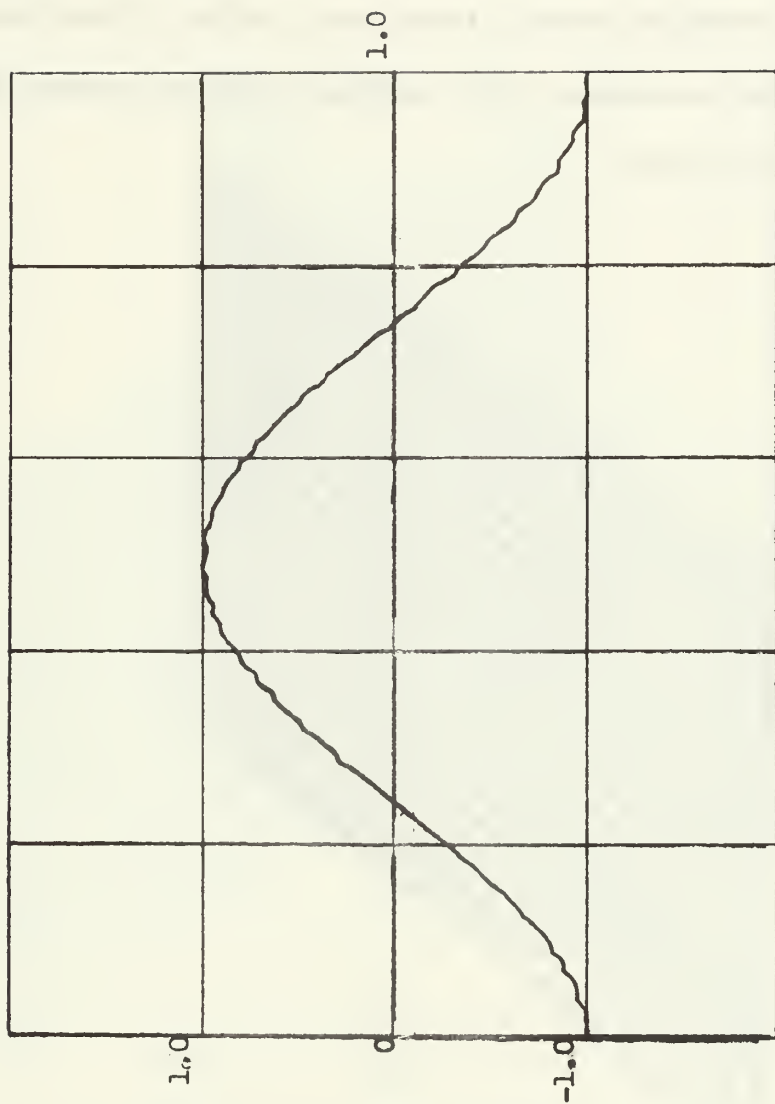
Special pressure distribution at $t = 6.50$ sec. of a matched
1-dimensional domain to a unit step input. Length=1.0 unit. $dx=0.01$



Horizontal scale = 2.00E-01 units/inch

Vertical scale = 1.00E+00 units/inch

Spatial pressure distribution at $t = 7.00$ sec. of a matched 1-dimensional domain to a unit step input. Length=1.0 unit. $dx=0.01$



Horizontal scale = 2.00E-01 units/inch

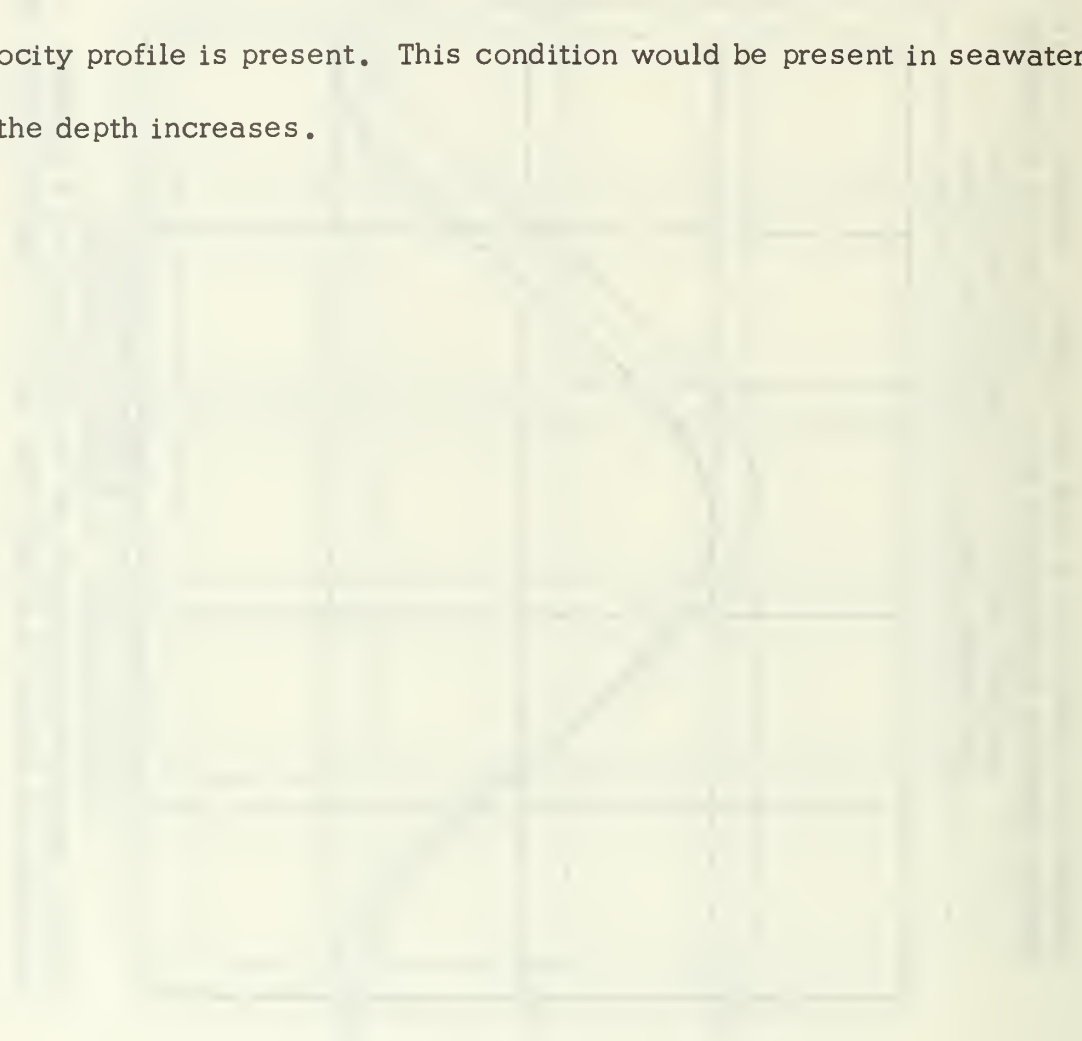
Vertical scale = 1.00E+00 units/inch

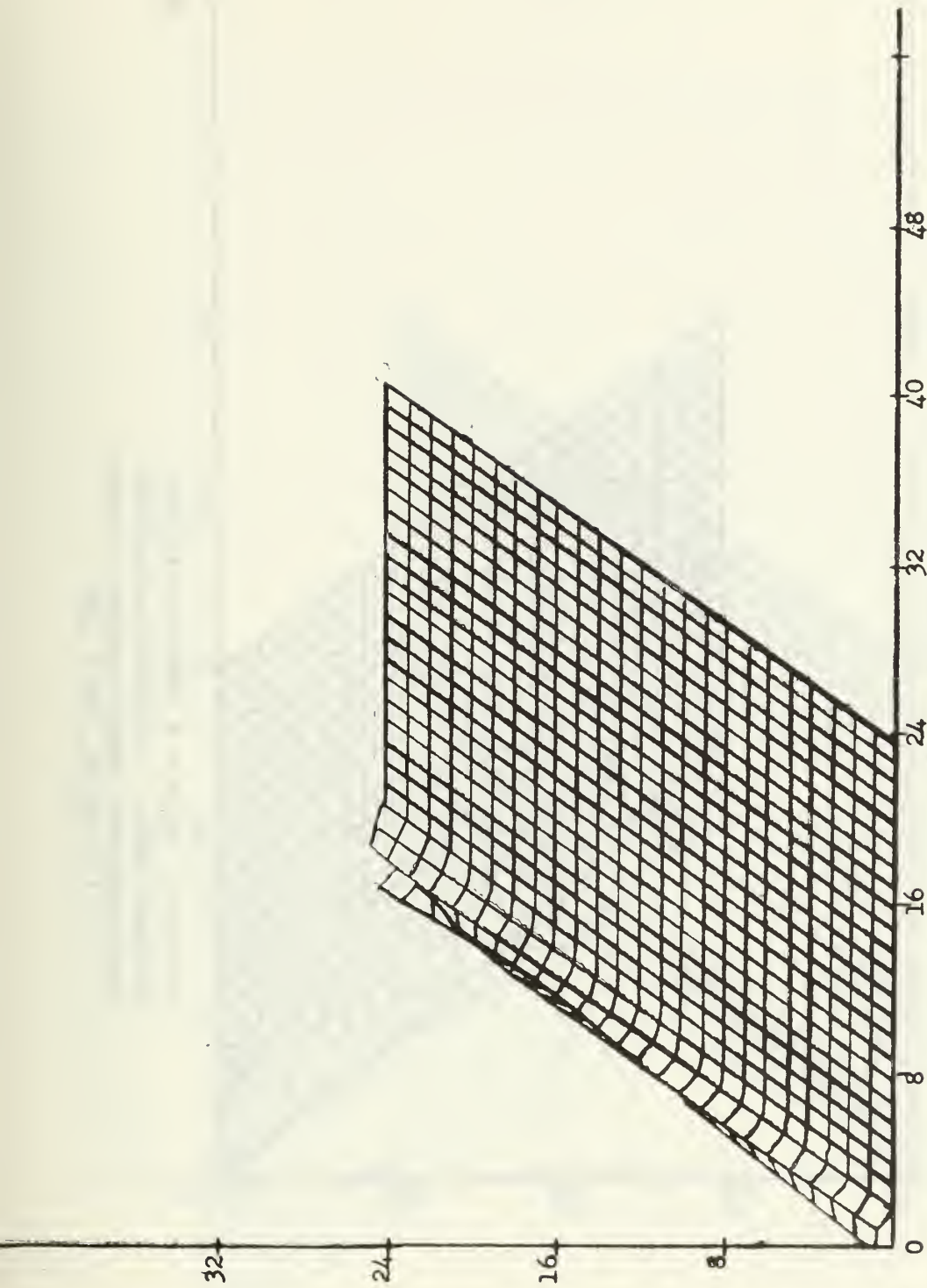
Spacial pressure distribution at $t = 7.50$ sec. of a matched 1-dimensional domain to a unit step input. Length=1.0 unit. $dx=0.01$

APPENDIX D

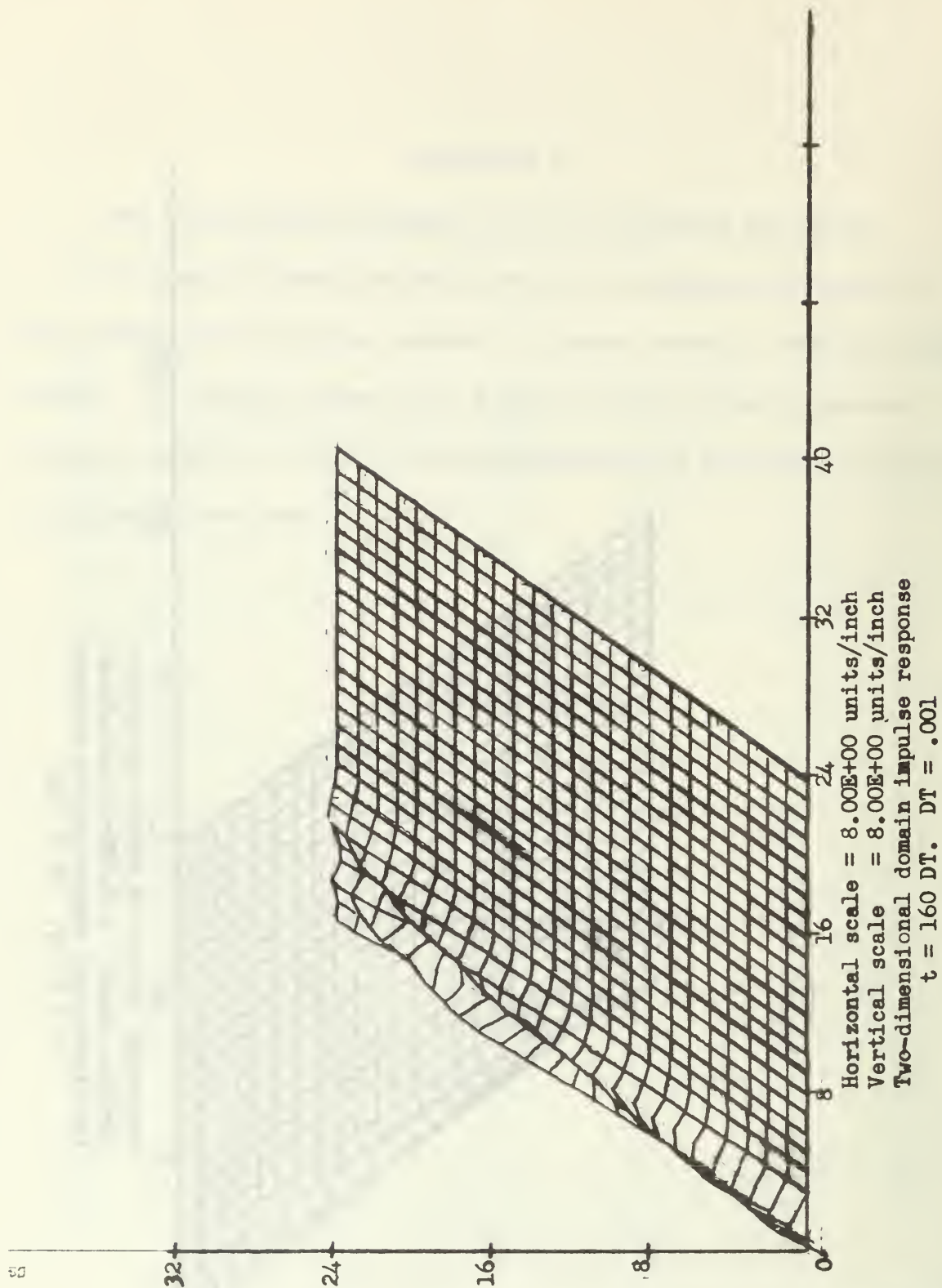
TWO-DIMENSIONAL VARIABLE-VELOCITY PROBLEM SOLUTION

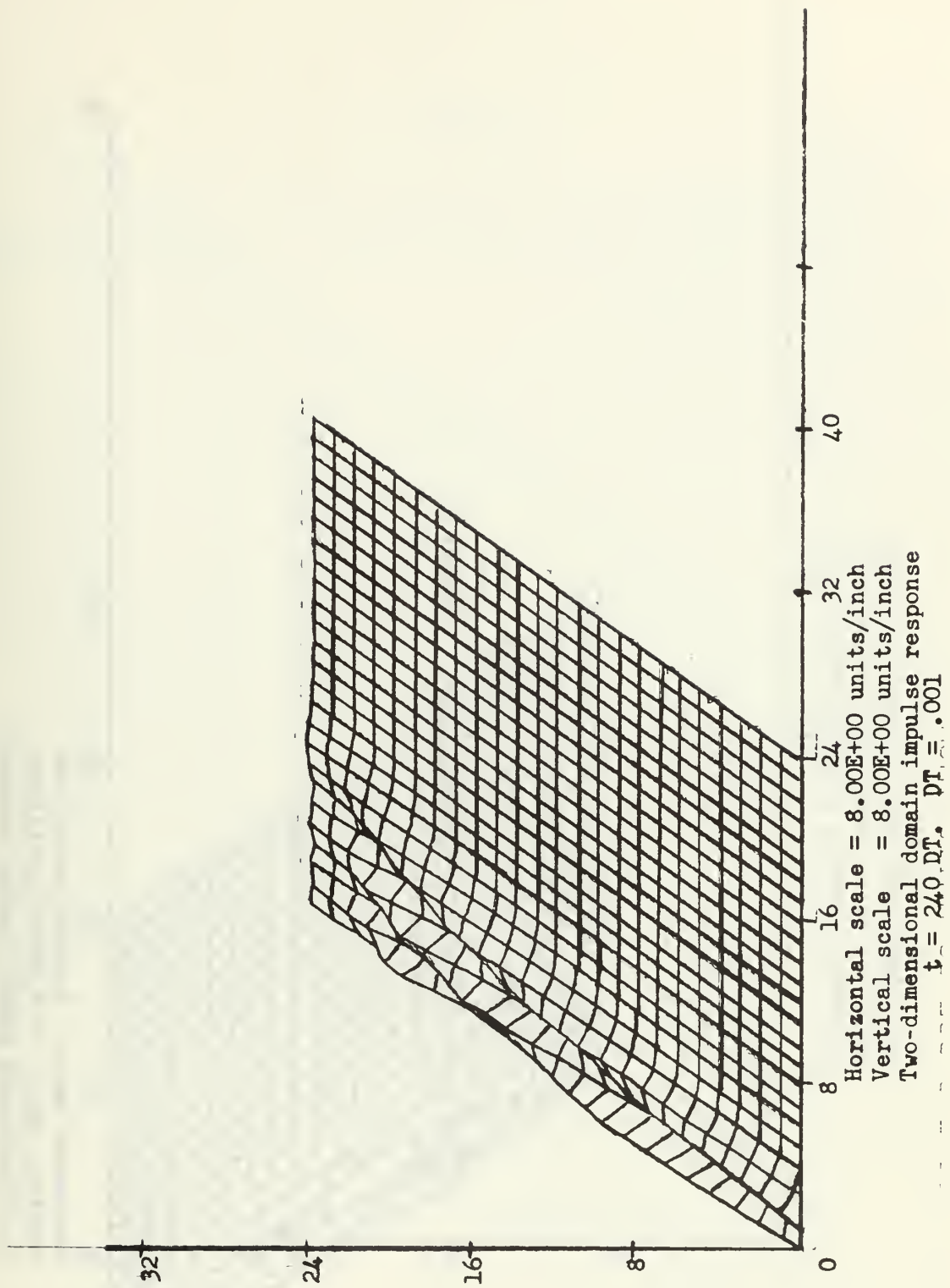
This appendix shows the two-dimensional pressure distribution for a line source initial condition applied to a medium having a variable-velocity profile. The example shown is for a case in which a linear increasing velocity profile is present. This condition would be present in seawater as the depth increases.

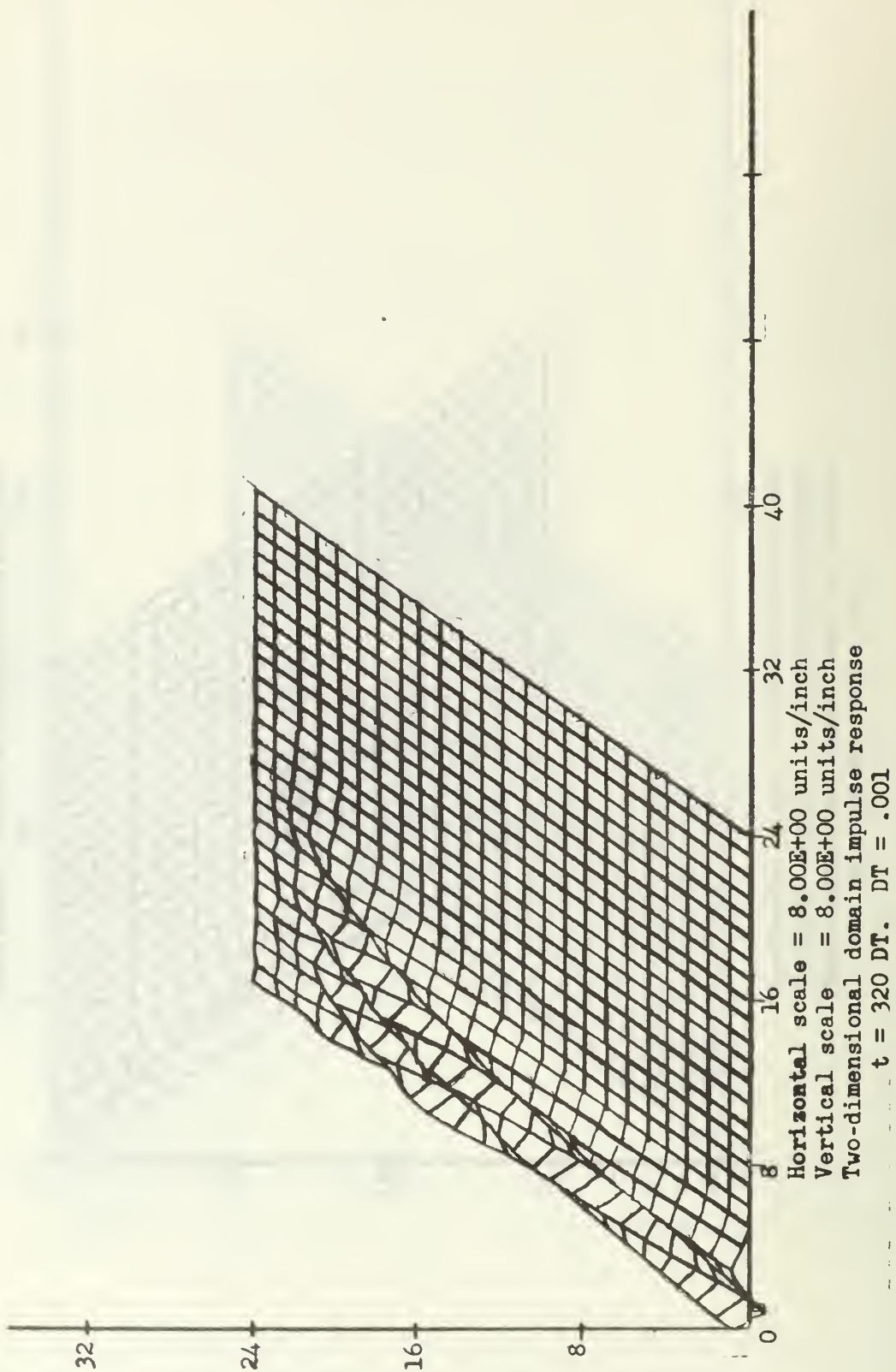


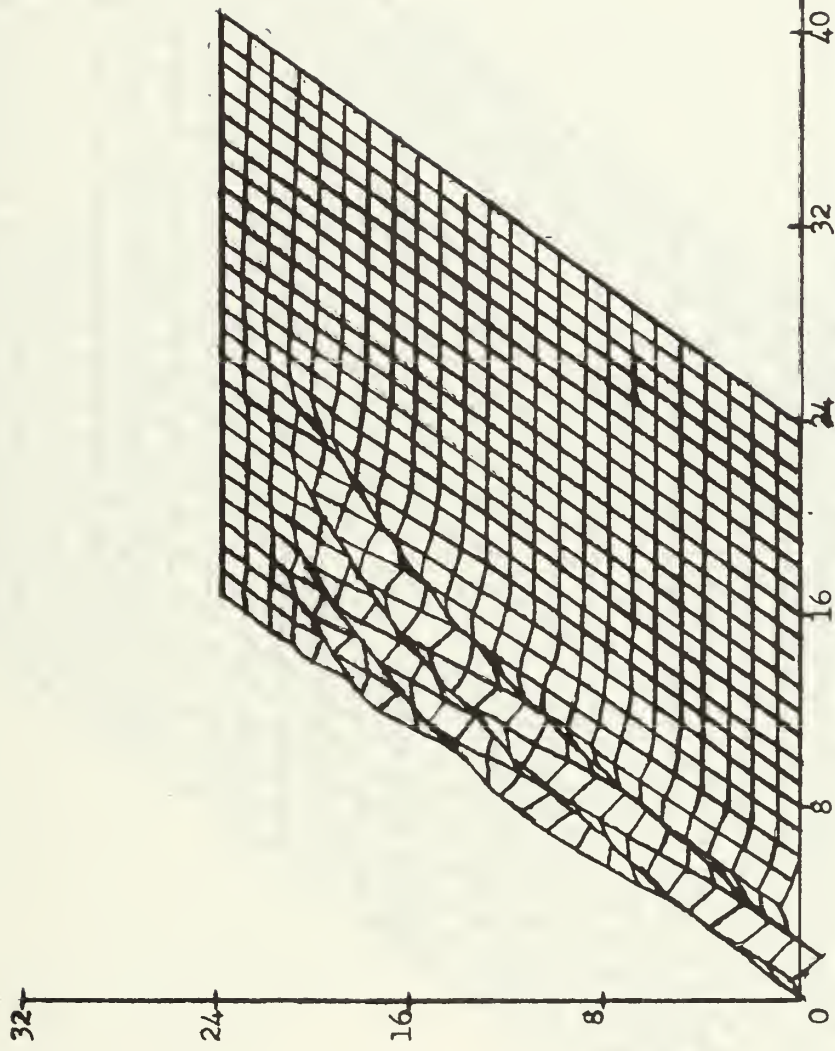


Horizontal scale = 8.00E+00 units/inch
 Vertical scale = 8.00E+00 units/inch
 Two-dimensional domain impulse response
 $t = 80 \text{ DT}$. $DT = .001$







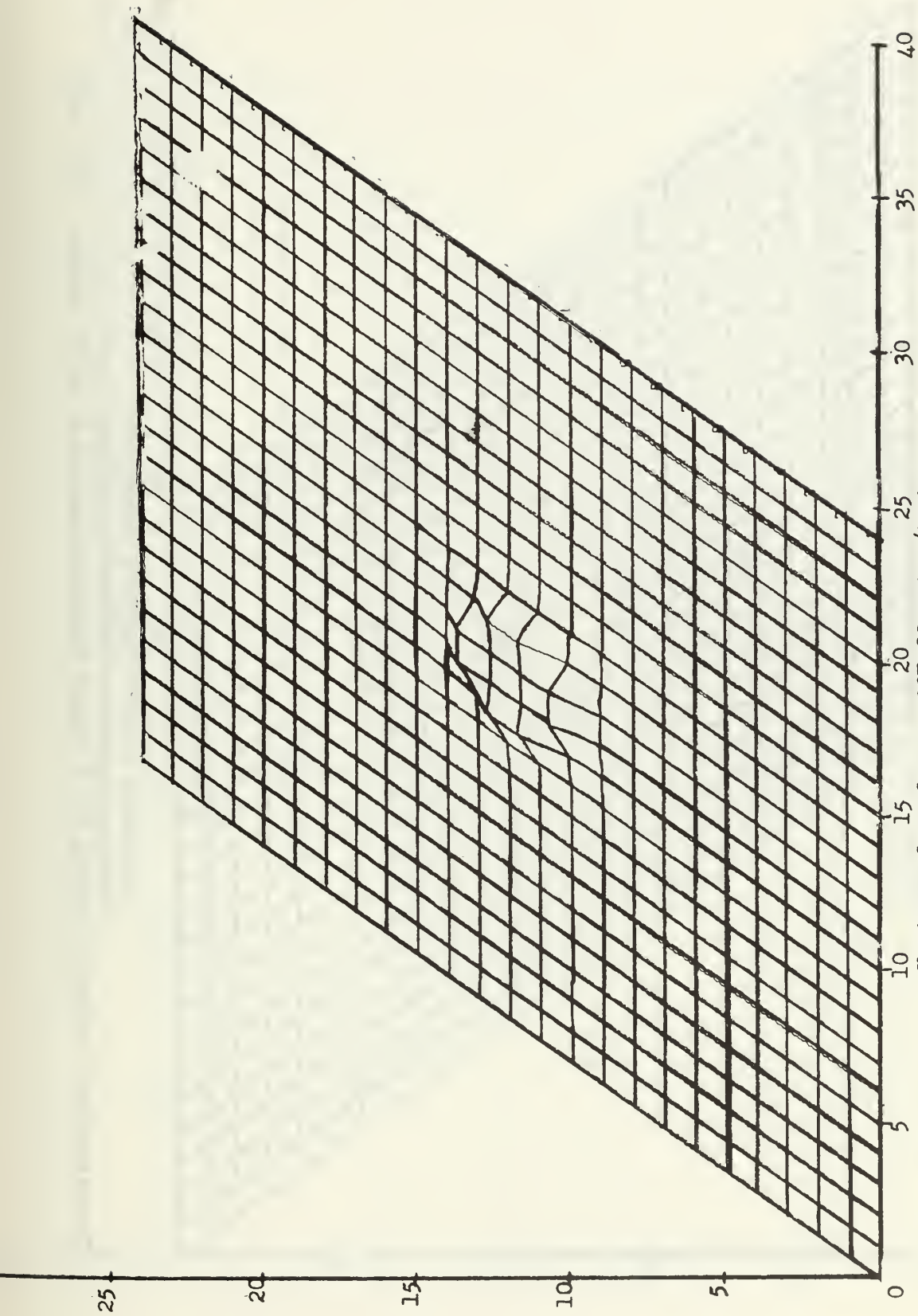


Horizontal scale = 8.00E+00 units/inch
 Vertical scale = 8.00E+00 units/inch
 Two-dimensional domain impulse response
 $t = 400 \text{ DT}$. $\text{DT} = .001$

APPENDIX E

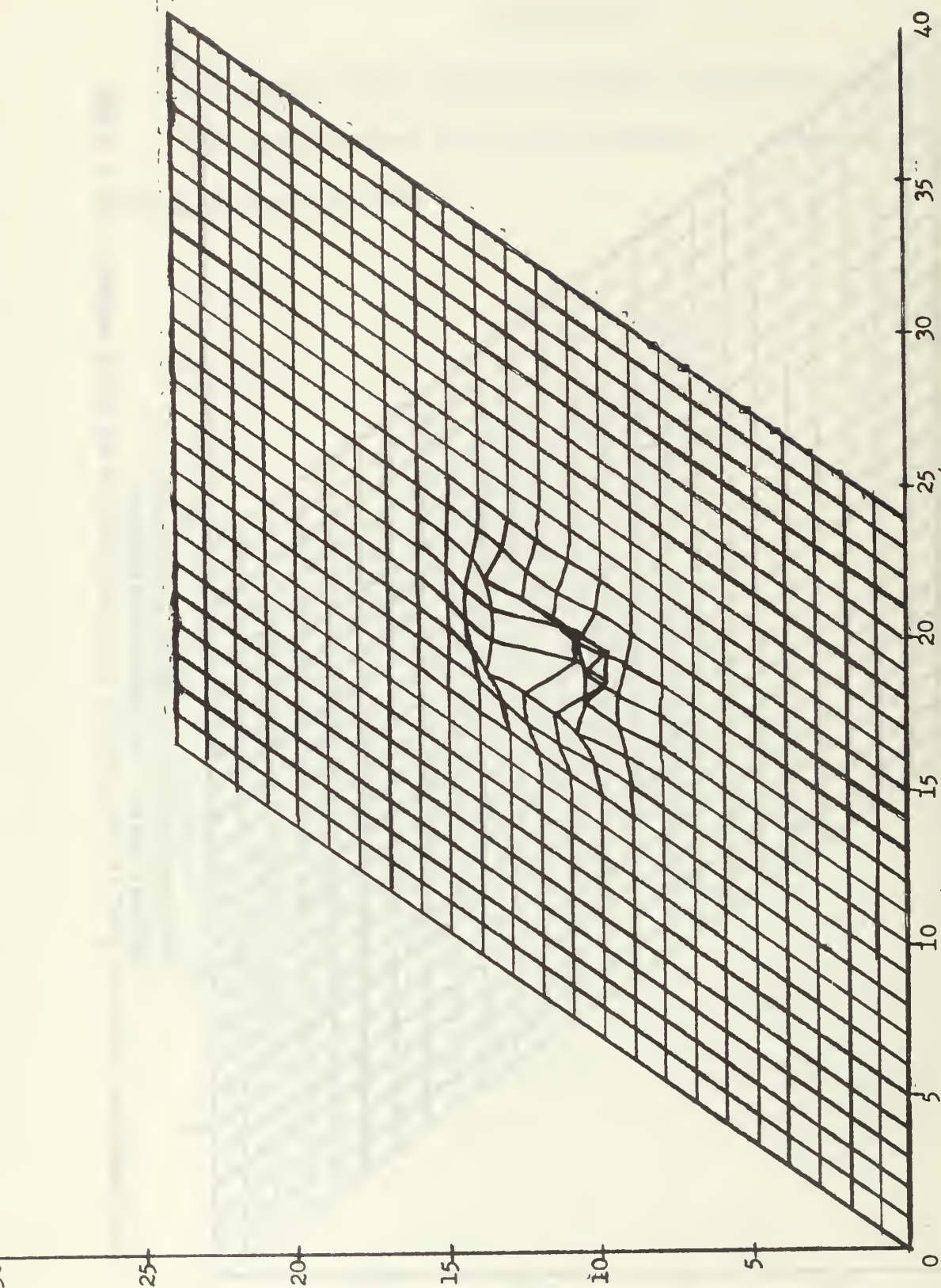
TWO-DIMENSIONAL NON-ZERO INITIAL CONDITION SOLUTION

This appendix shows the transient solution to a deformed two-dimensional membrane problem.



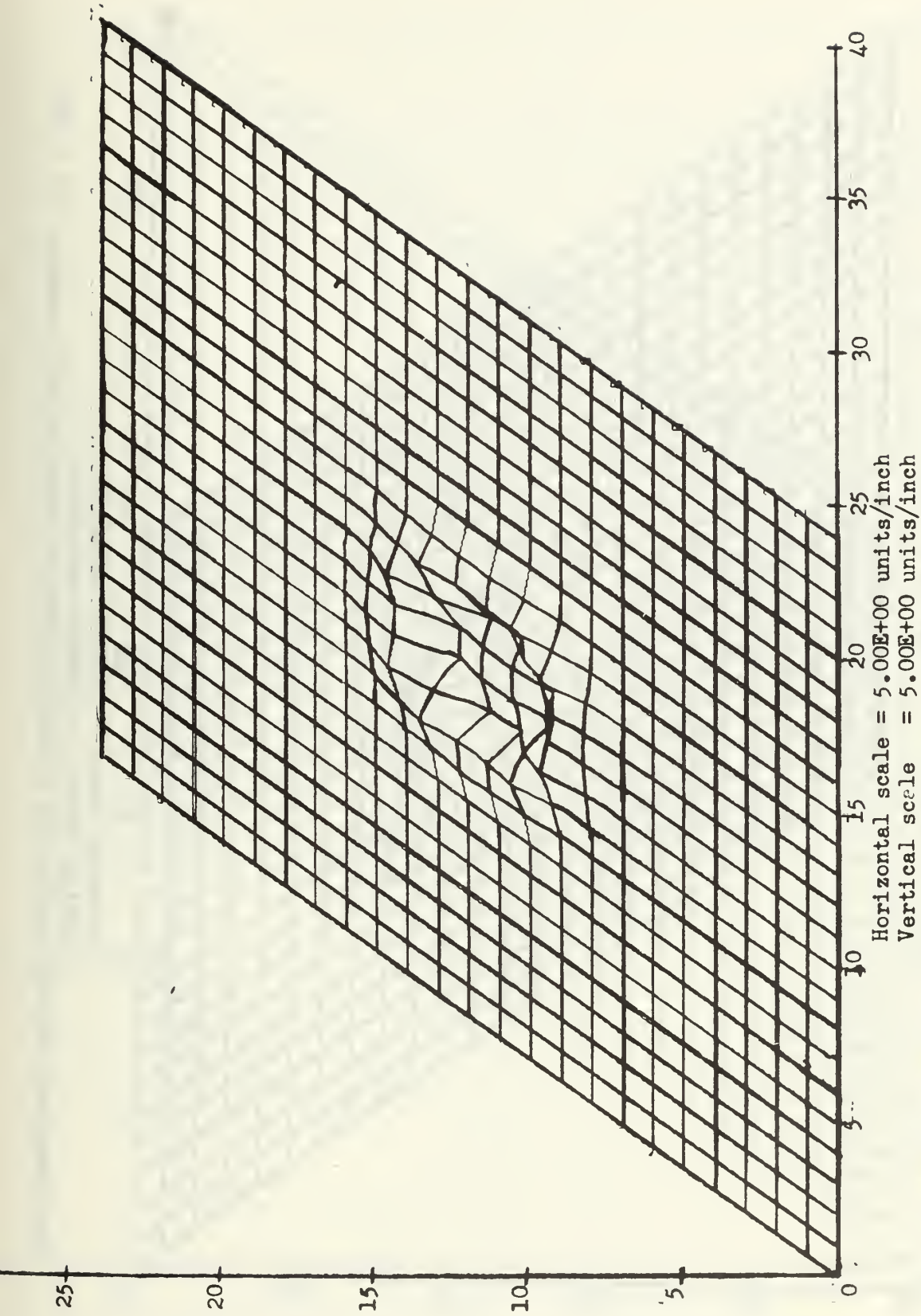
Horizontal scale = 5.00E+00 units/inch
 Vertical scale = 5.00E+00 units/inch

Two-dimensional velocity profile for a highly dispersive medium at 20 DT seconds. DT = 0.001



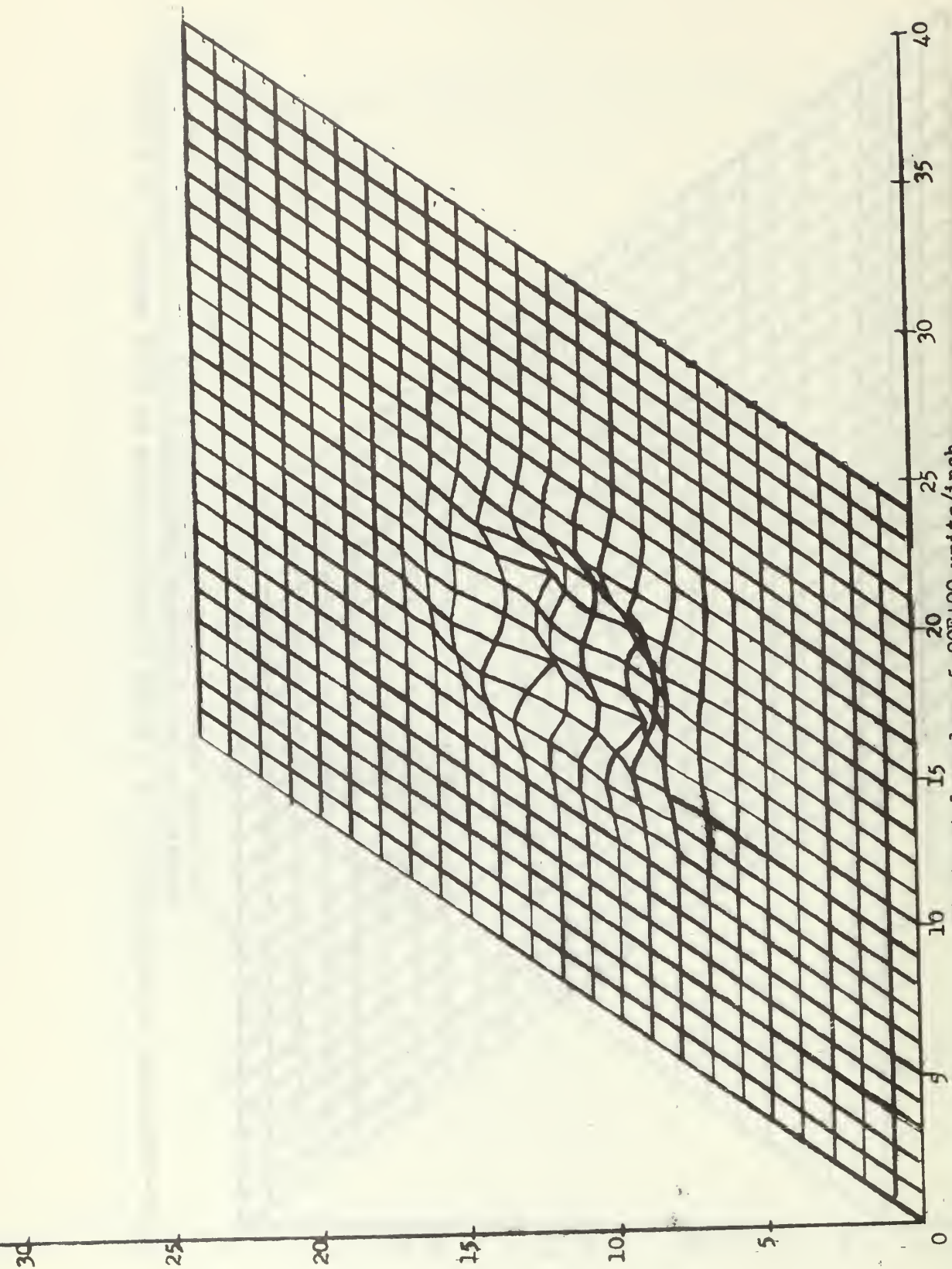
Horizontal scale = 5.00E+00 units/inch
Vertical scale = 5.00E+00 units/inch

Two-dimensional velocity profile for a highly dispersive medium at 40 DT seconds. $DT = 0.001$



Horizontal scale = 5.00E+00 units/inch
Vertical scale = 5.00E+00 units/inch

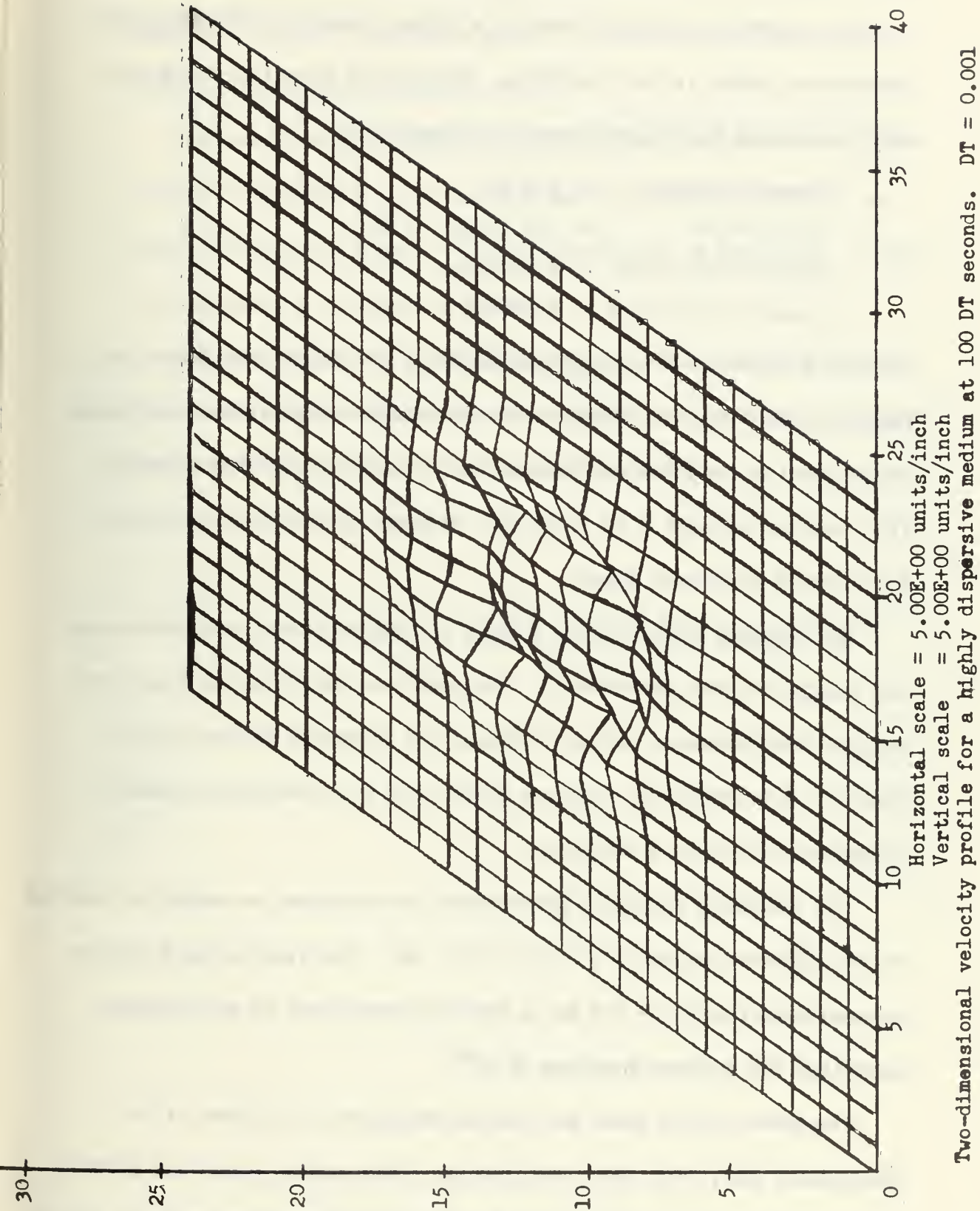
Two-dimensional velocity profile for a highly dispersive medium at 60 DT seconds. DT = 0.001



Horizontal scale = 5.00E+00 units/inch

Vertical scale = 5.00E+00 units/inch

Two-dimensional velocity profile for a highly dispersive medium at 80 DT seconds. DT = .001



APPENDIX F

The graphical results of running a Widrow adaptive filter simulation program are shown in this appendix. The desired signal to which the array responded had the following characteristics:

center frequency = 500 Hz
band-width = 200 Hz and 0 Hz
desired look angle = 45 degrees
power signal input = 1.0 watt

The noise signal is of standard deviation 0.707 and is provided by the random number program GUASS; thus for wideband signals the total noise power input to the array may exceed that of the signal before filtering. The noise is incident at an angle of 0 degrees relative to the normal of the plane of the linear array.

The filtering action occurs in both in frequency response selectivity and spacial angular selectivity. The graphs on the following pages are grouped into frequency response graphs and receiving pattern graphs. There are four frequency response graphs and four receiving patterns shown for each time of adaption.

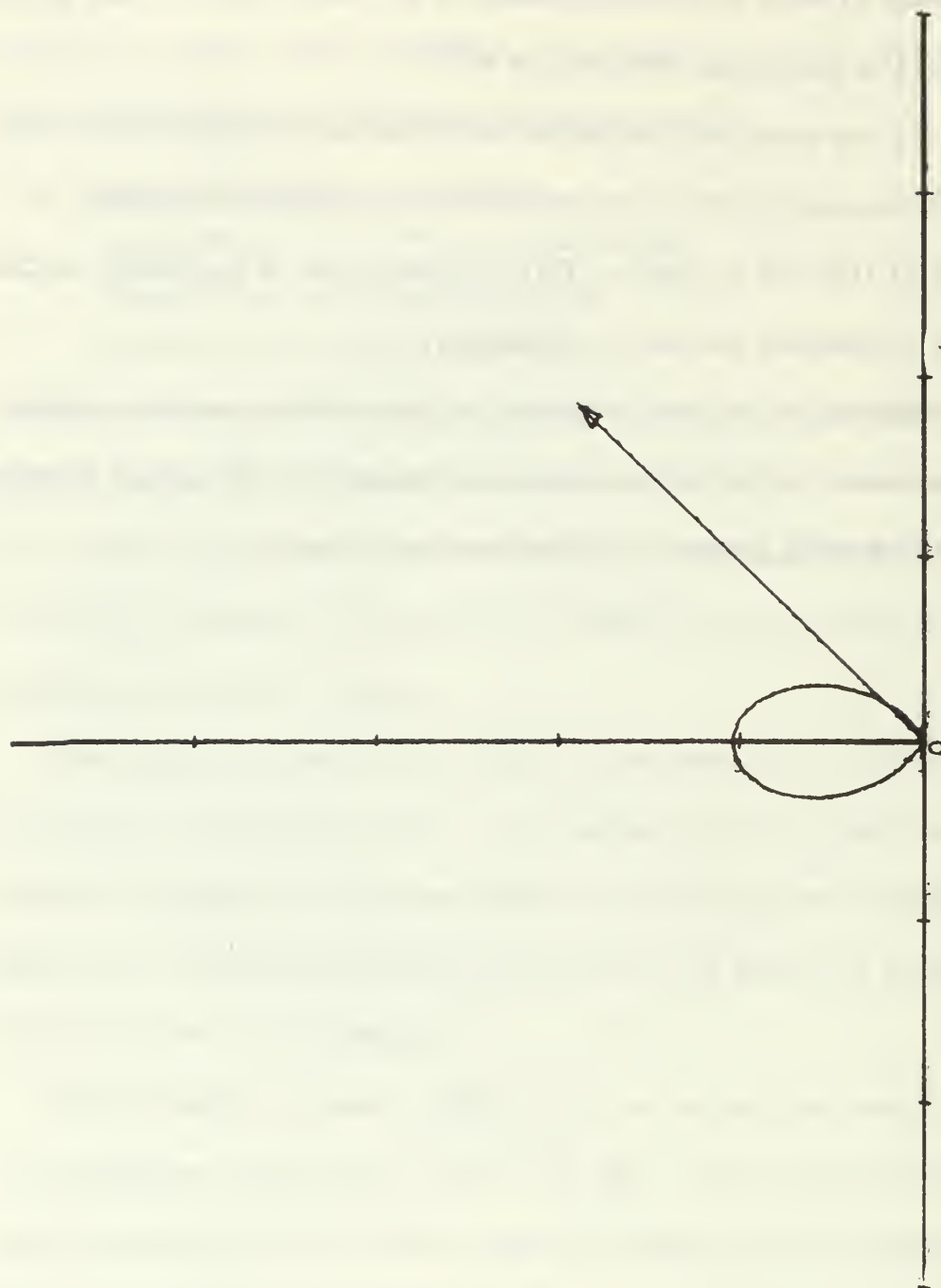
The frequency response graphs show the response as would be observed for the different angles of 0° , 30° , 45° , 90° . The results show that the desired signal of 500 ± 100 Hz is strongly attenuated for all directions other than the desired direction of 45° .

The pattern plots show the spacial response of the filter at the frequencies 100, 200, 500, and 1000 Hz. The results show that there is greatly reduced array pattern gain for the desired frequency of 500 ± 100 Hz

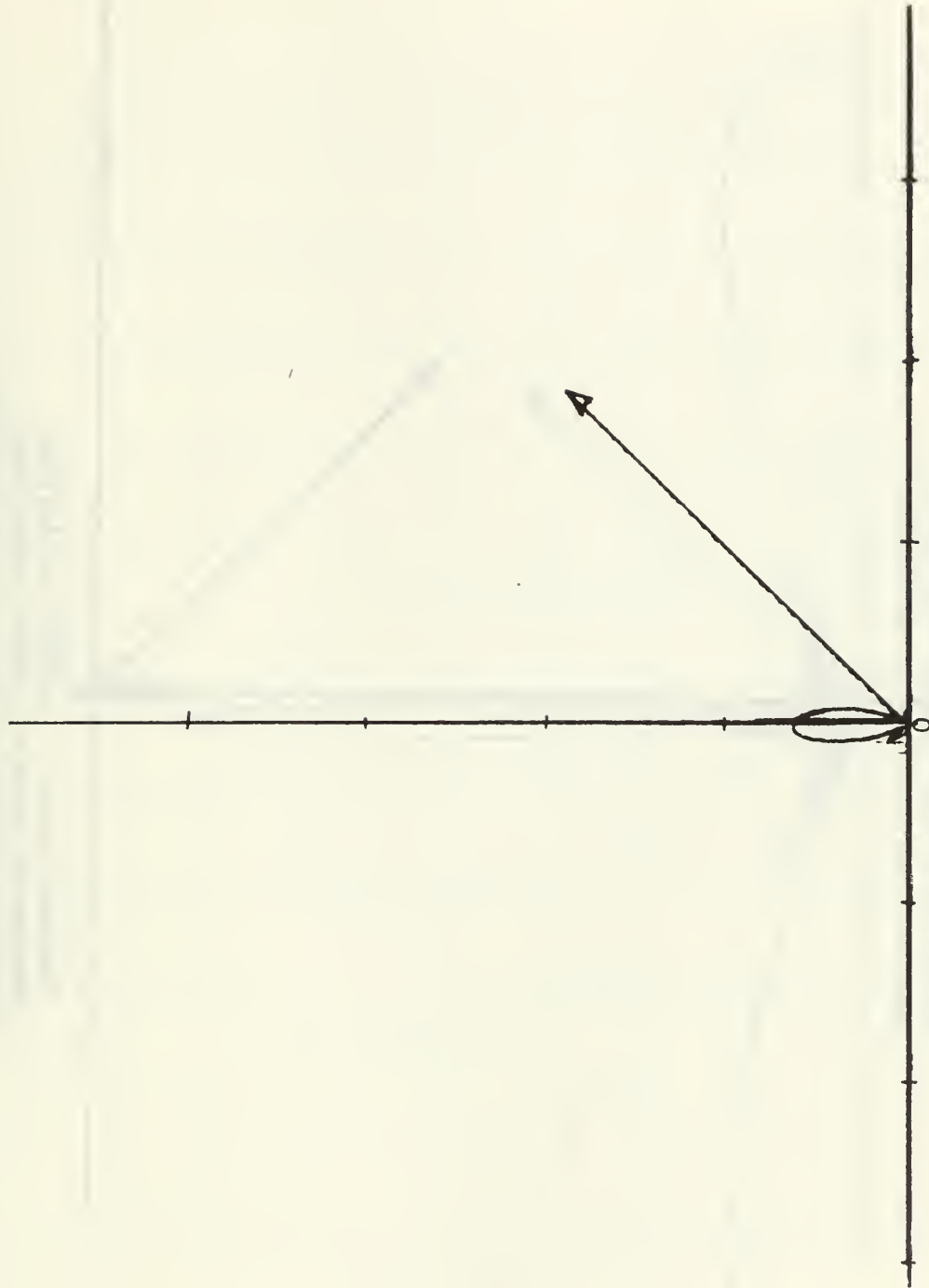
in all directions except the design direction of 45° . In addition, the response of array to the frequencies of 100 Hz and 200 Hz are greatly reduced in the design direction of 45° .

The one prominent exception to the desired behavior is that the filter has appreciable gain and frequency response for signals or noise at 1000 Hz or higher. This is correctable as mentioned in Chap. VI by appropriate low-pass prefiltering.

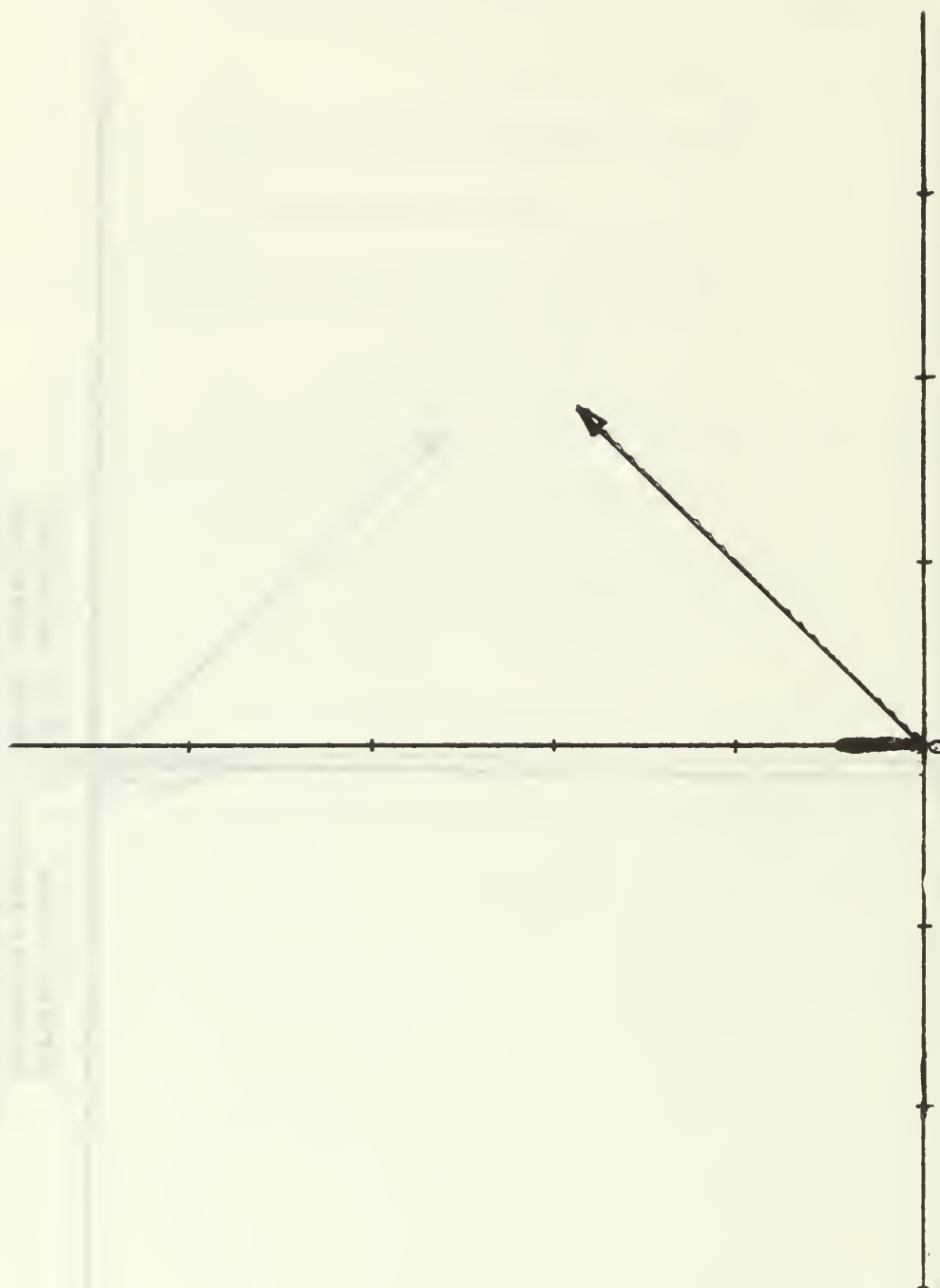
The results for zero bandwidth desired signals are very close to those shown for the wide-band signal except that the rate of adaption is considerably greater in the desired look direction.



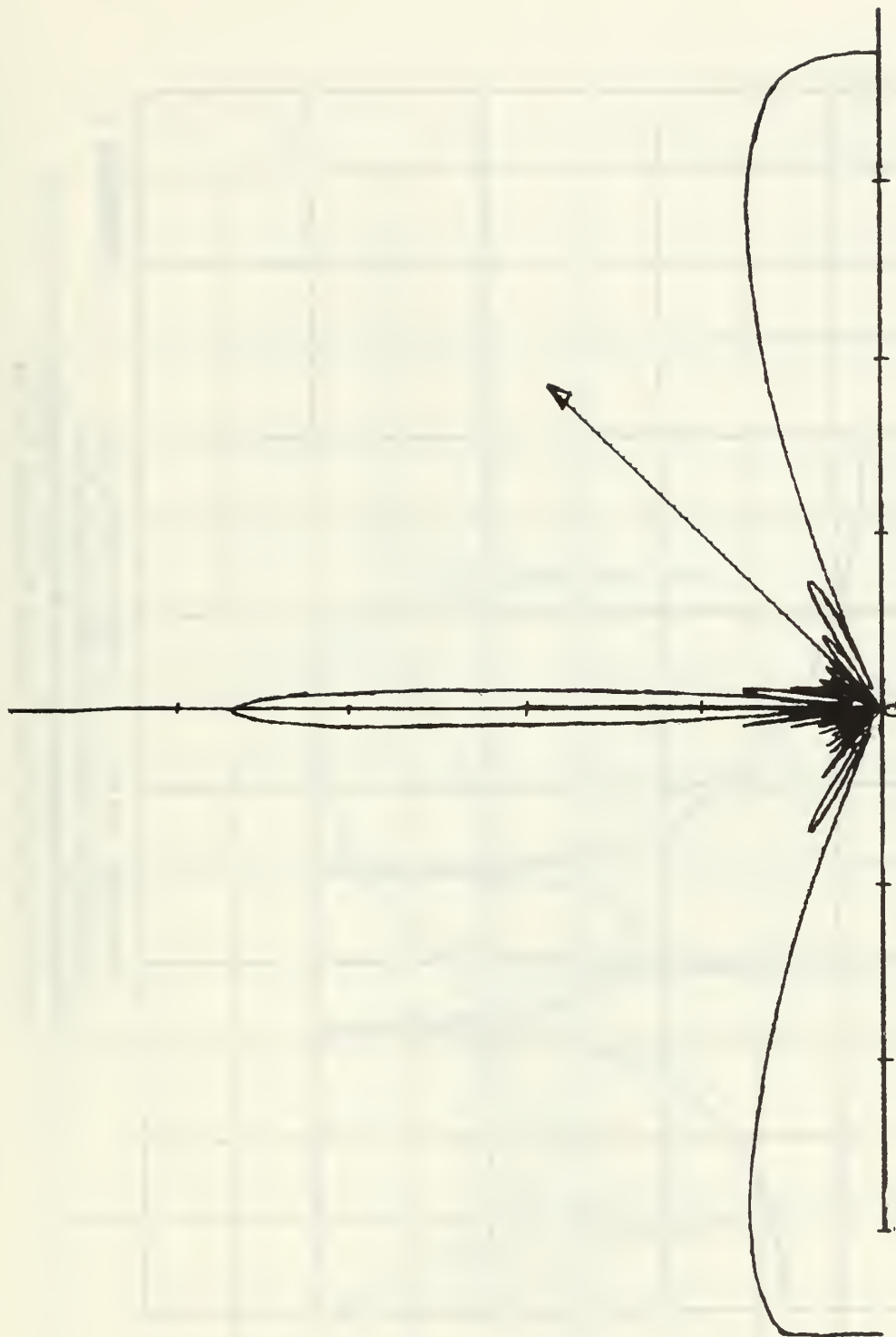
Vertical scale = 4.00E-01 units/inch
Horizontal scale = 4.00E-01 units/inch
Directivity pattern at 100 Hz after adapting
for 15 cycles of the desired frequency 500 Hz.



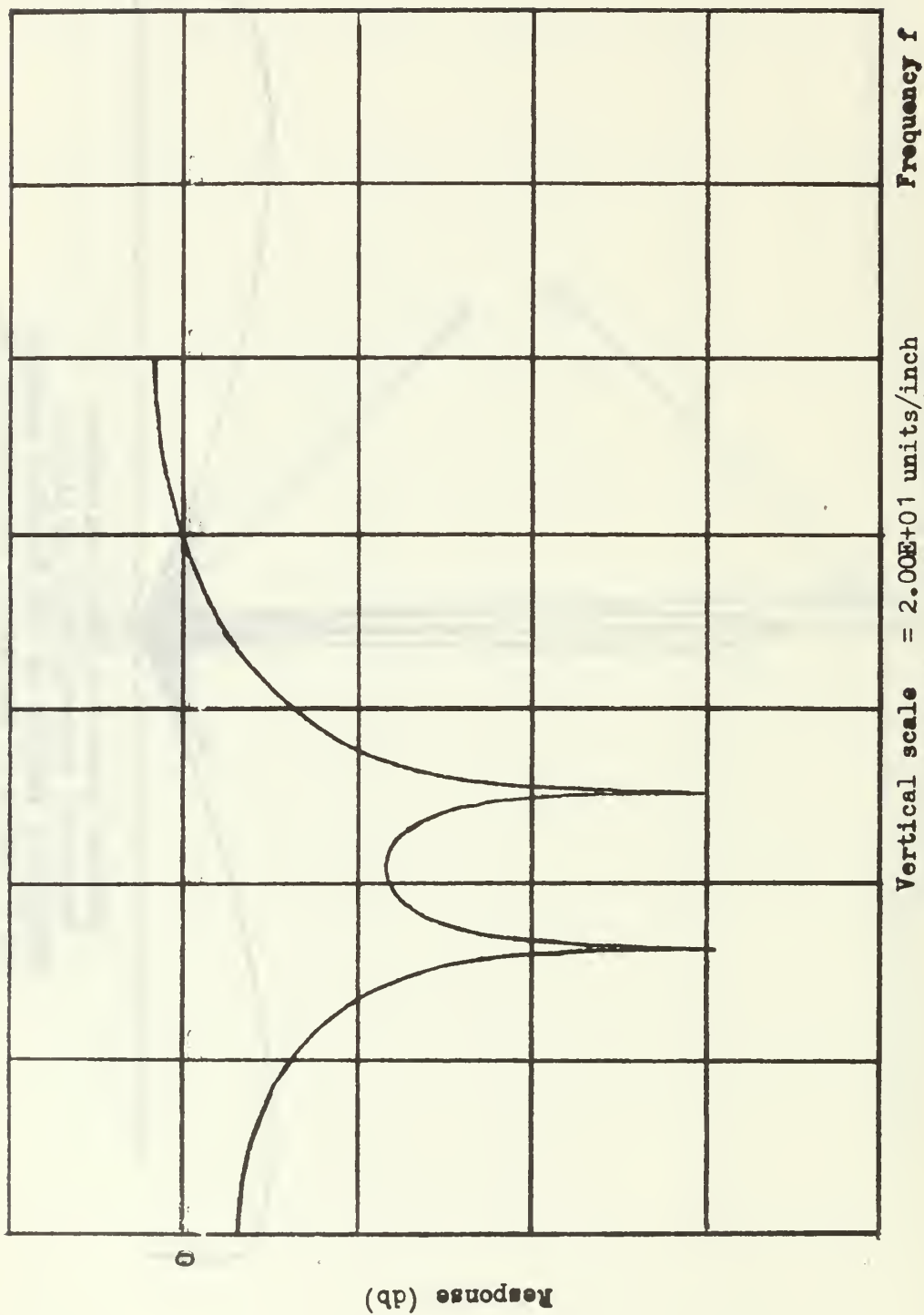
Vertical scale = 4.00E-01 units/inch
Horizontal scale = 4.00E-01 units/inch
Directivity pattern at 200 Hz after adapting
for 15 cycles of the desired frequency 500 Hz.



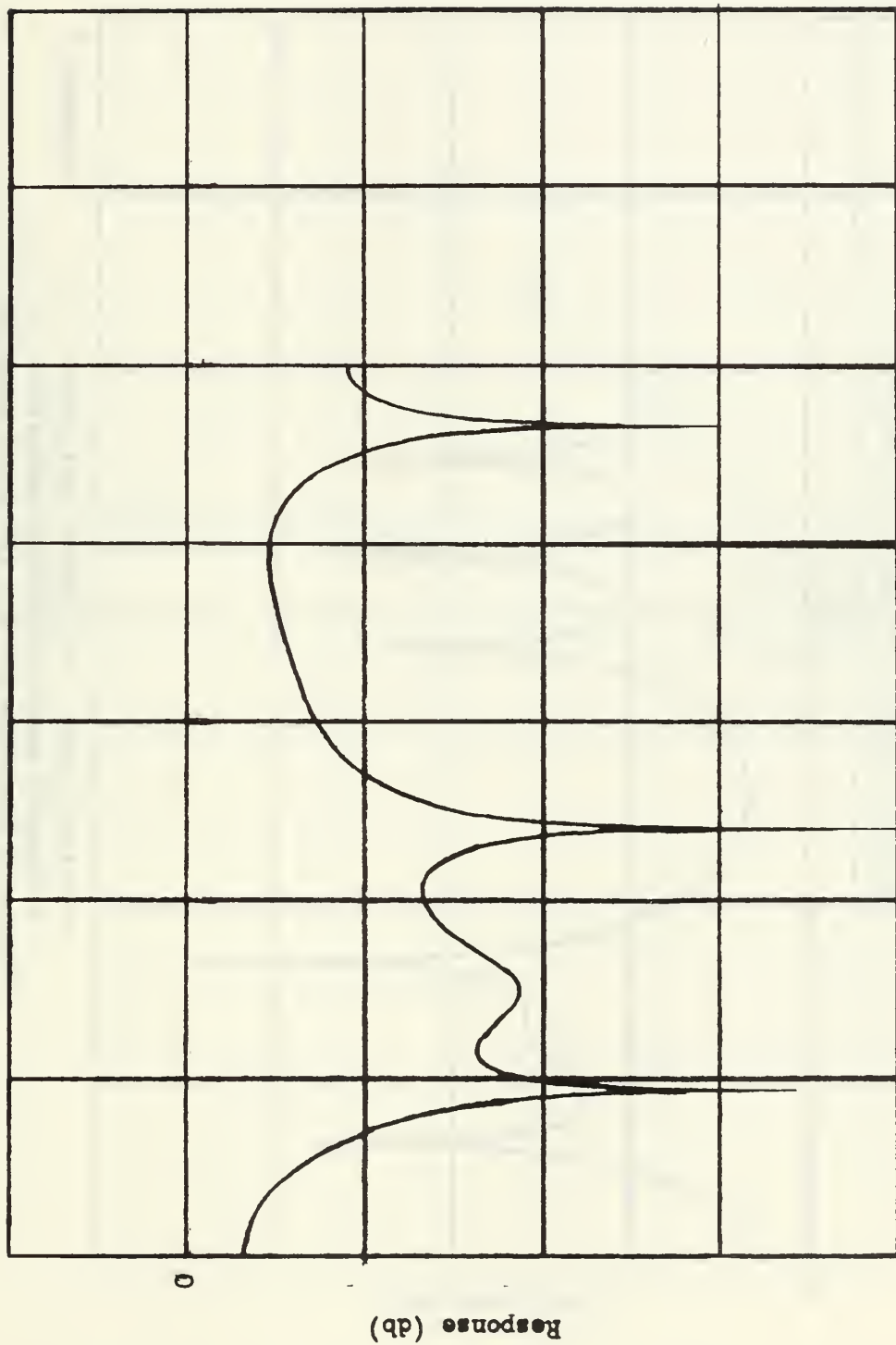
Vertical scale = 4.00E-01 units/inch
Horizontal scale = 4.00E-01 units/inch
Directivity pattern at 500 Hz after adapting
for 15 cycles of the desired frequency 500 Hz.



Vertical scale = 4.00E-01 units/inch
Horizontal scale=4.00E-01 units/inch
Directivity pattern at 1000 Hz after adapting
for 15 cycles of the desired frequency 500 Hz.



Frequency responses for $\theta = 0^\circ$ after adapting for 15 cycles of the desired frequency 500 Hz.

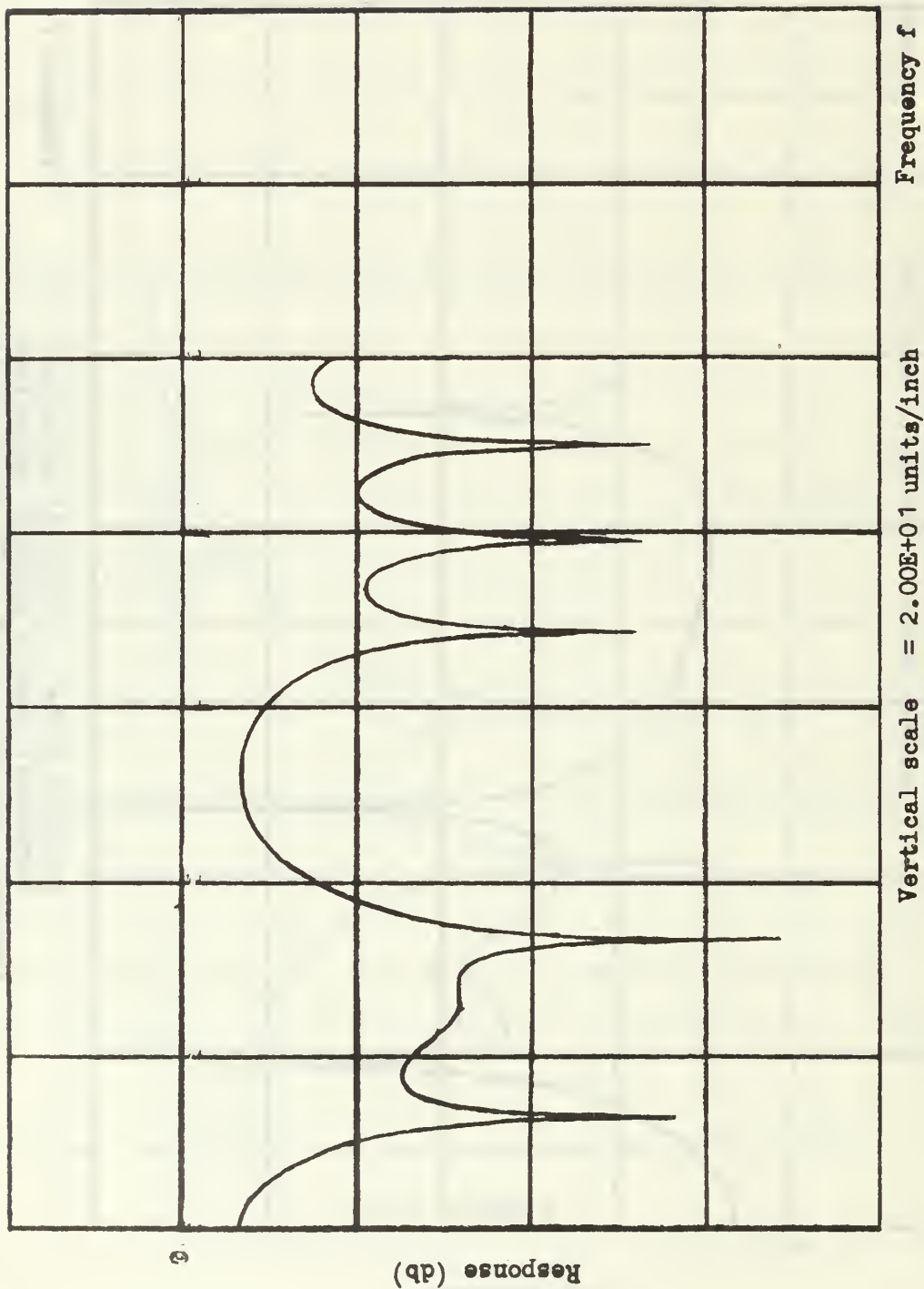


Frequency f

Vertical scale = $2.00E+01$ units/inch

Horizontal scale = $2.00E+02$ units/inch

Frequency responses for $\theta = 30^\circ$ after adapting for 15 cycles of the desired frequency 500 Hz.

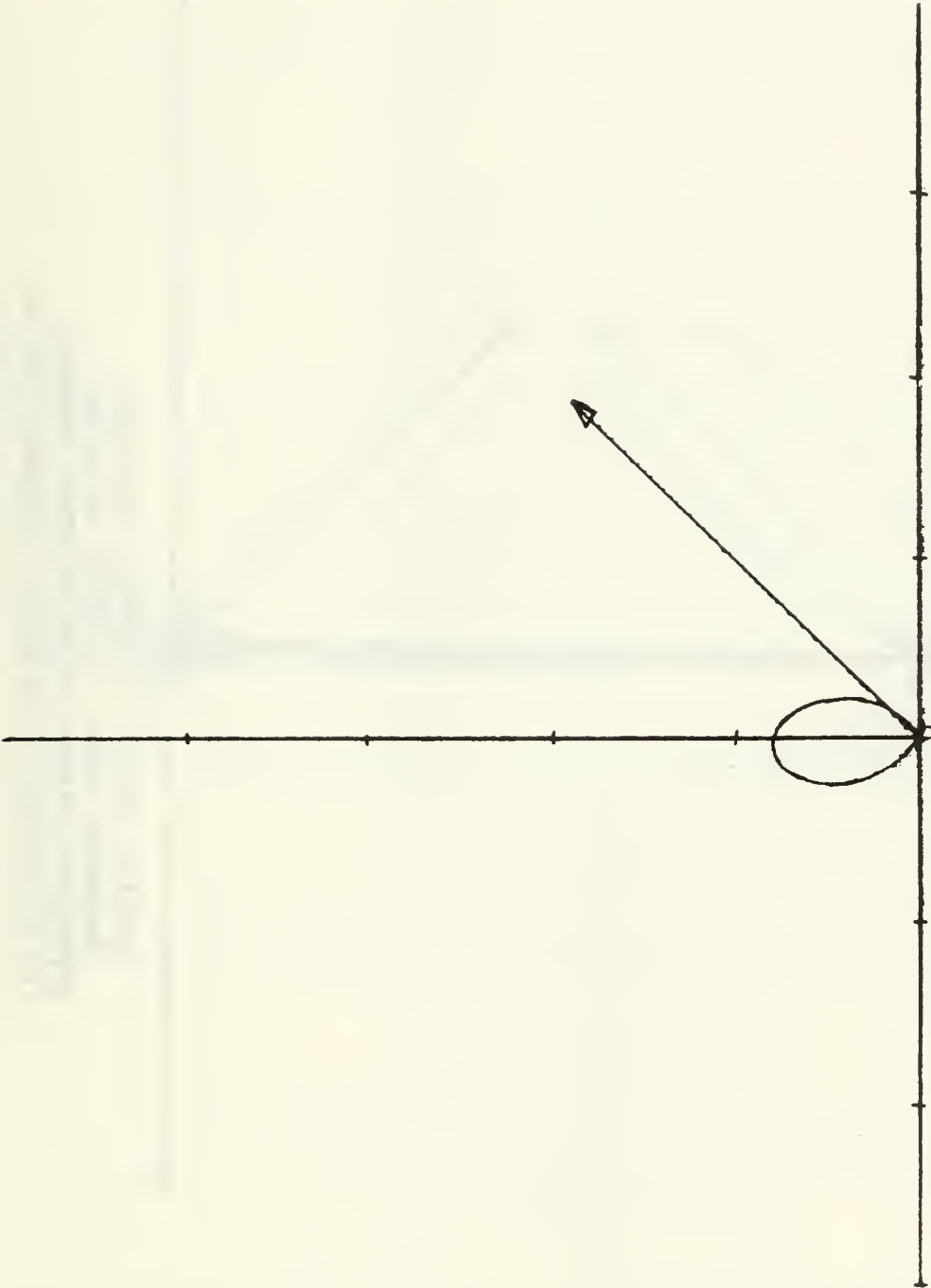


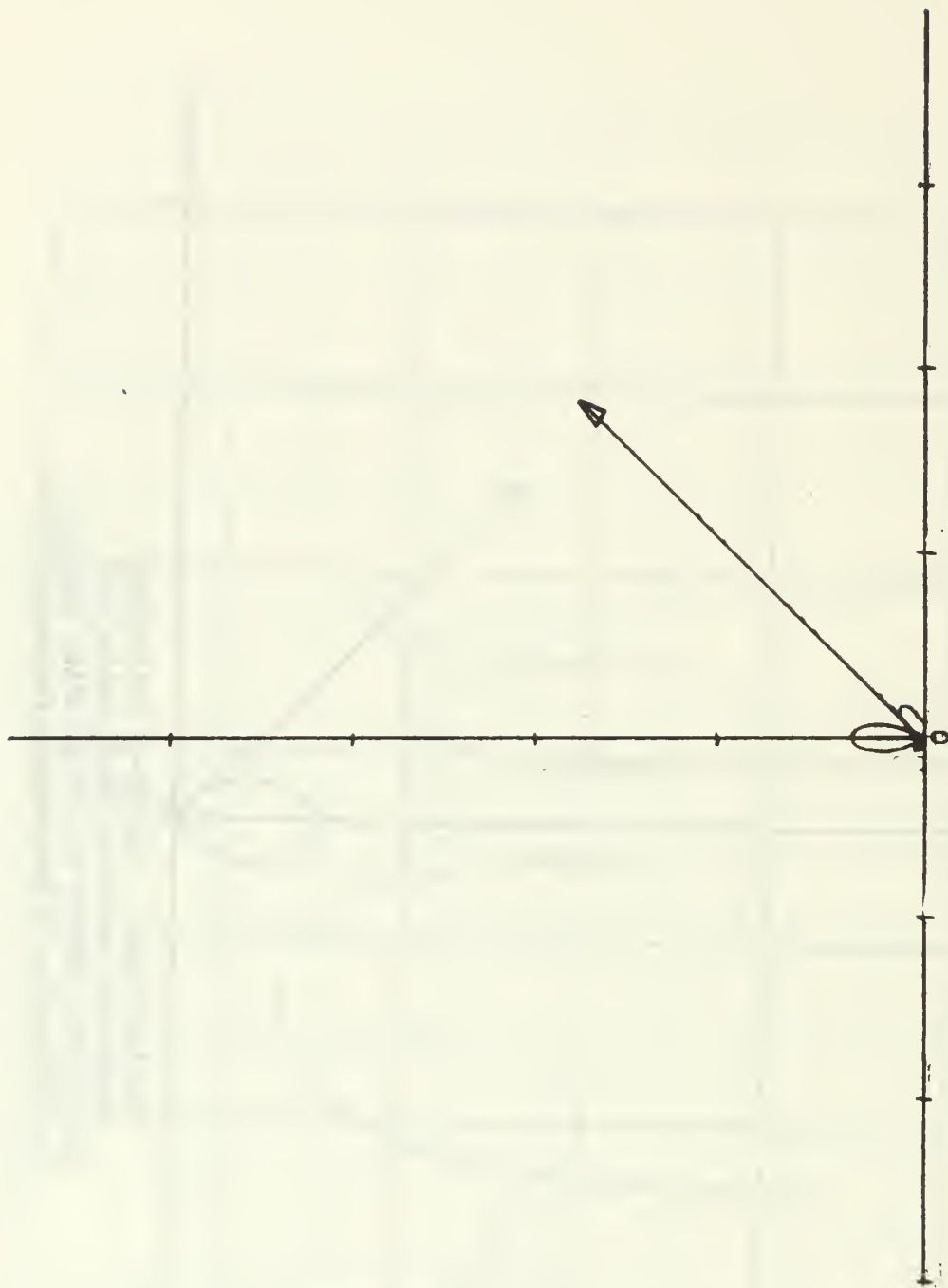
Frequency f

Vertical scale = $2.00E+01$ units/inch

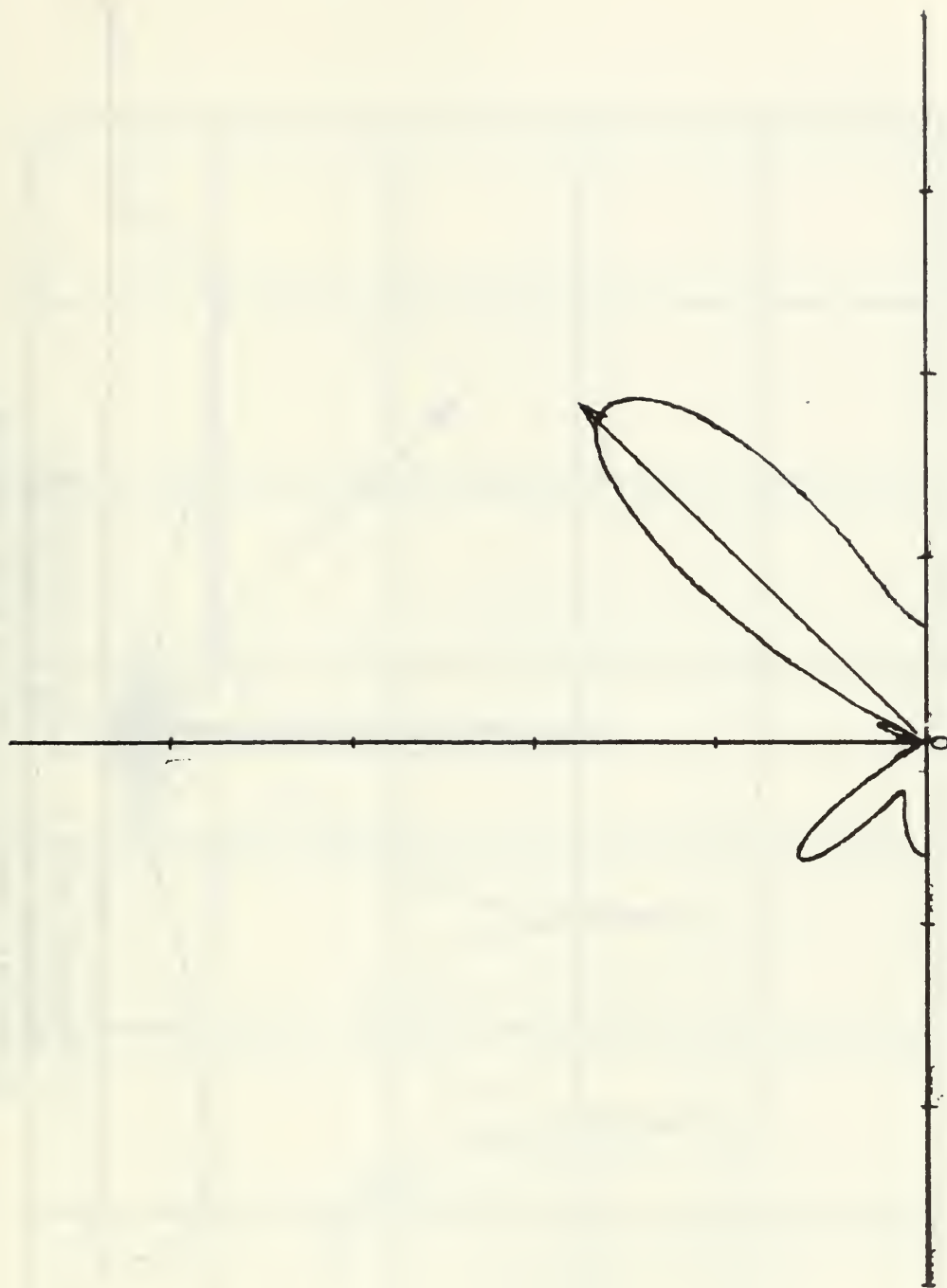
Horizontal scale = $2.00E+02$ units/inch

Frequency responses for $\theta = 45^\circ$ after adapting for 15 cycles of the desired frequency 500 Hz.

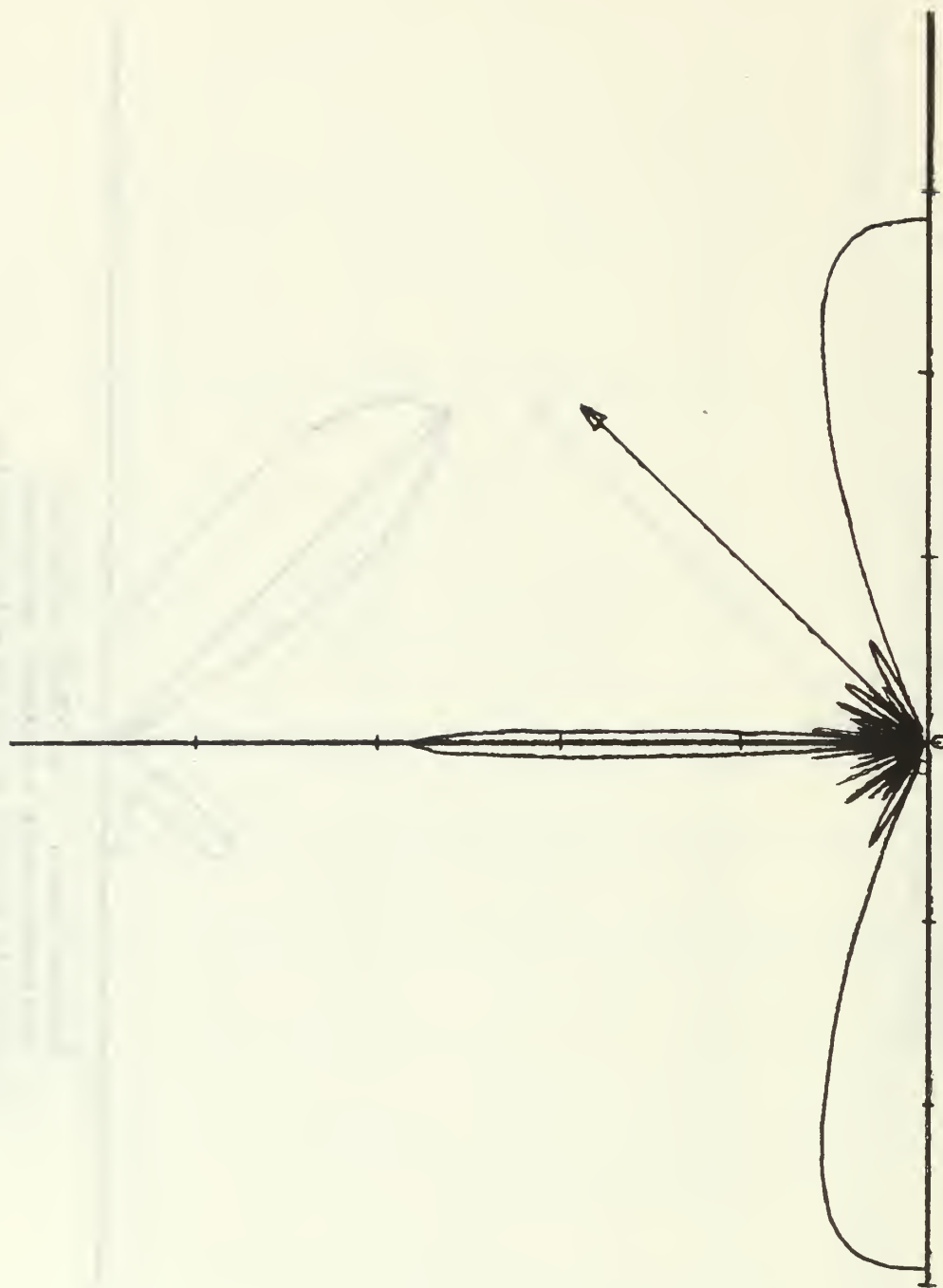




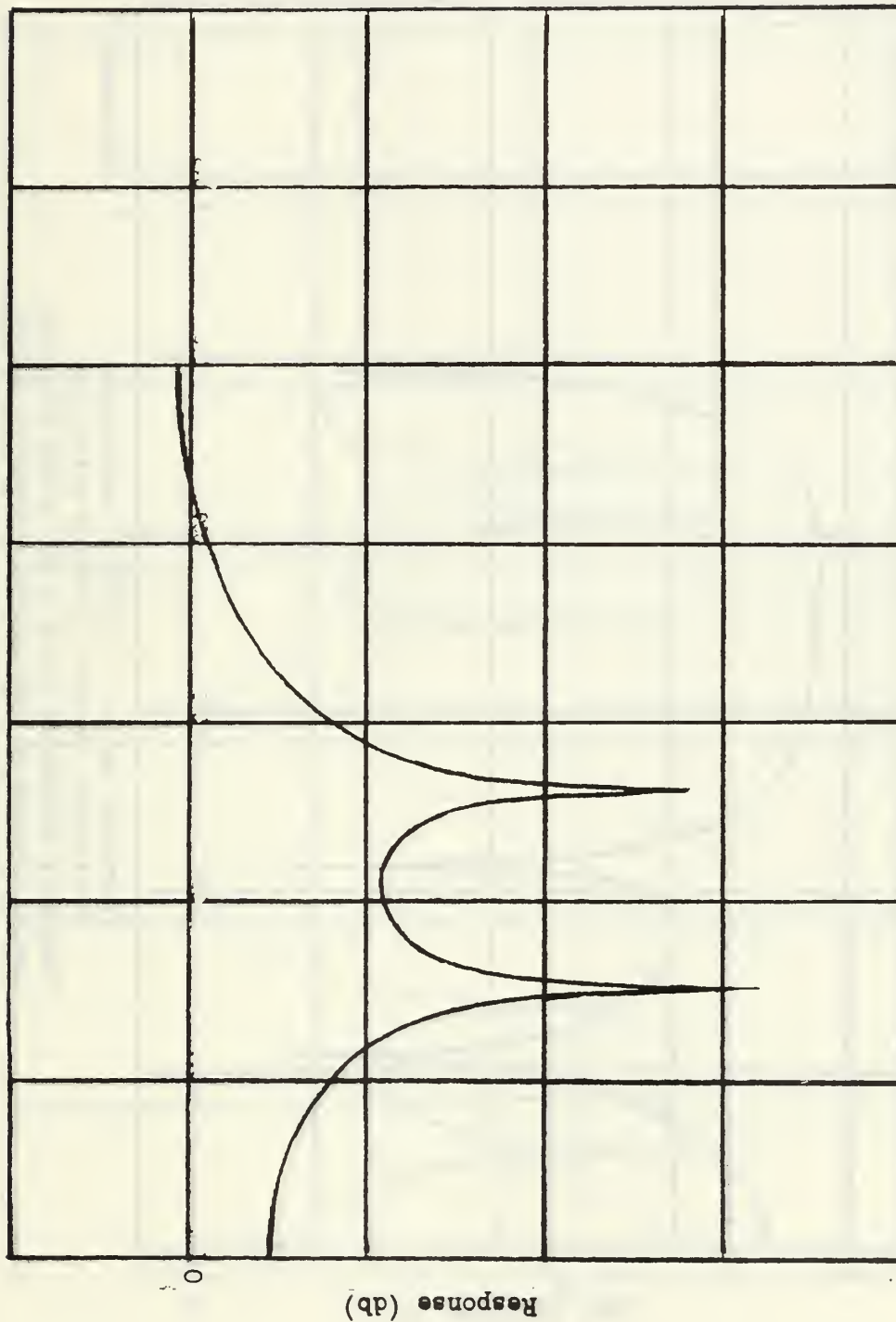
Vertical scale = 4.00E-01 units/inch
Horizontal scale = 4.00E-01 units/inch
Directivity pattern at 200 Hz after adapting
for 300 cycles of the desired frequency 500 Hz.



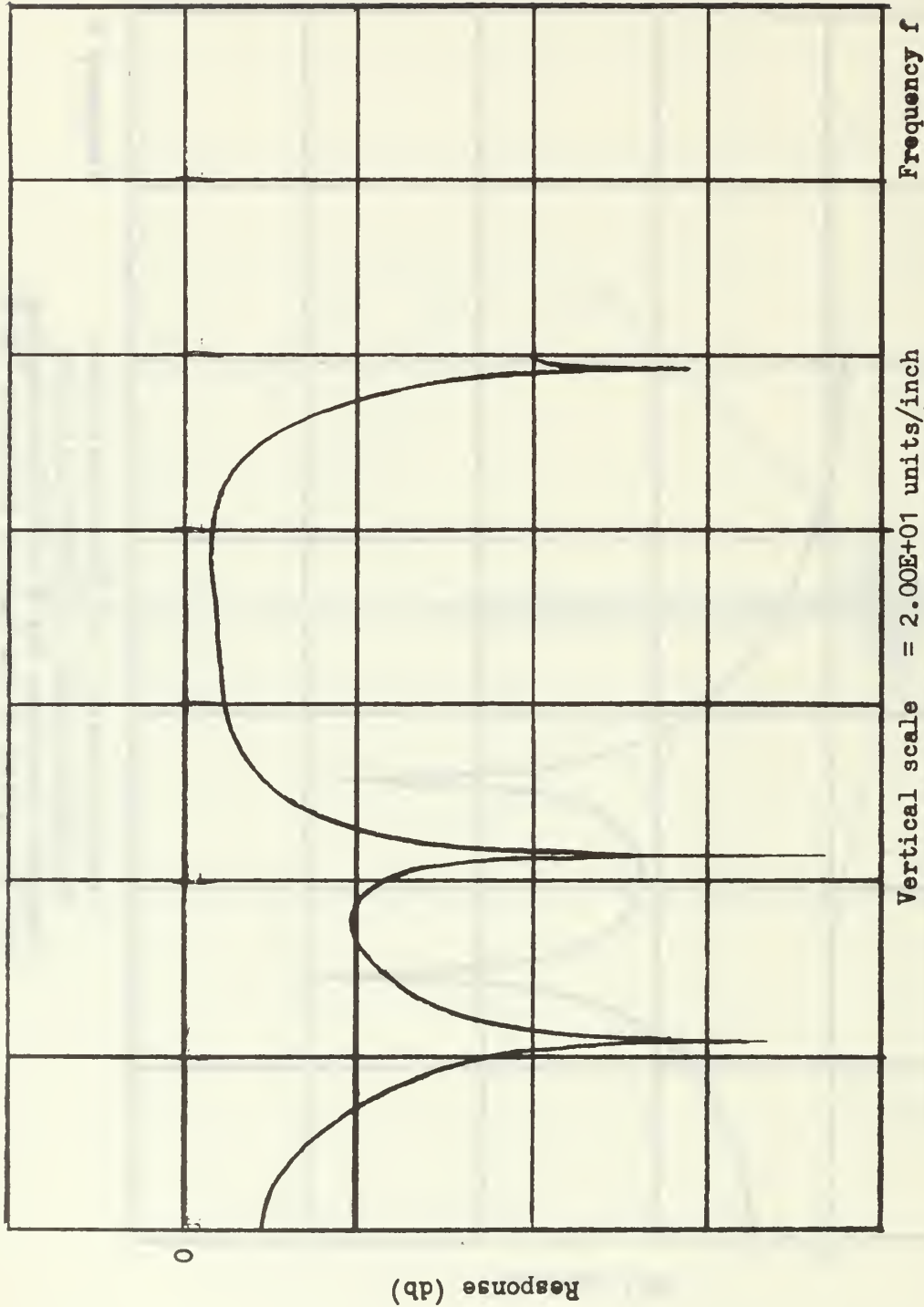
Vertical scale = 4.00E-01 units/inch
Horizontal scale = 4.00E-01 units/inch
Directivity pattern at 500 Hz after adapting
for 300 cycles of the desired frequency 500 Hz.



Vertical scale = 4.00E-01 units/inch
Horizontal scale = 4.00E-01 units/inch
Directivity pattern at 1000 Hz after adapting
for 300 cycles of the desired frequency 500 Hz.



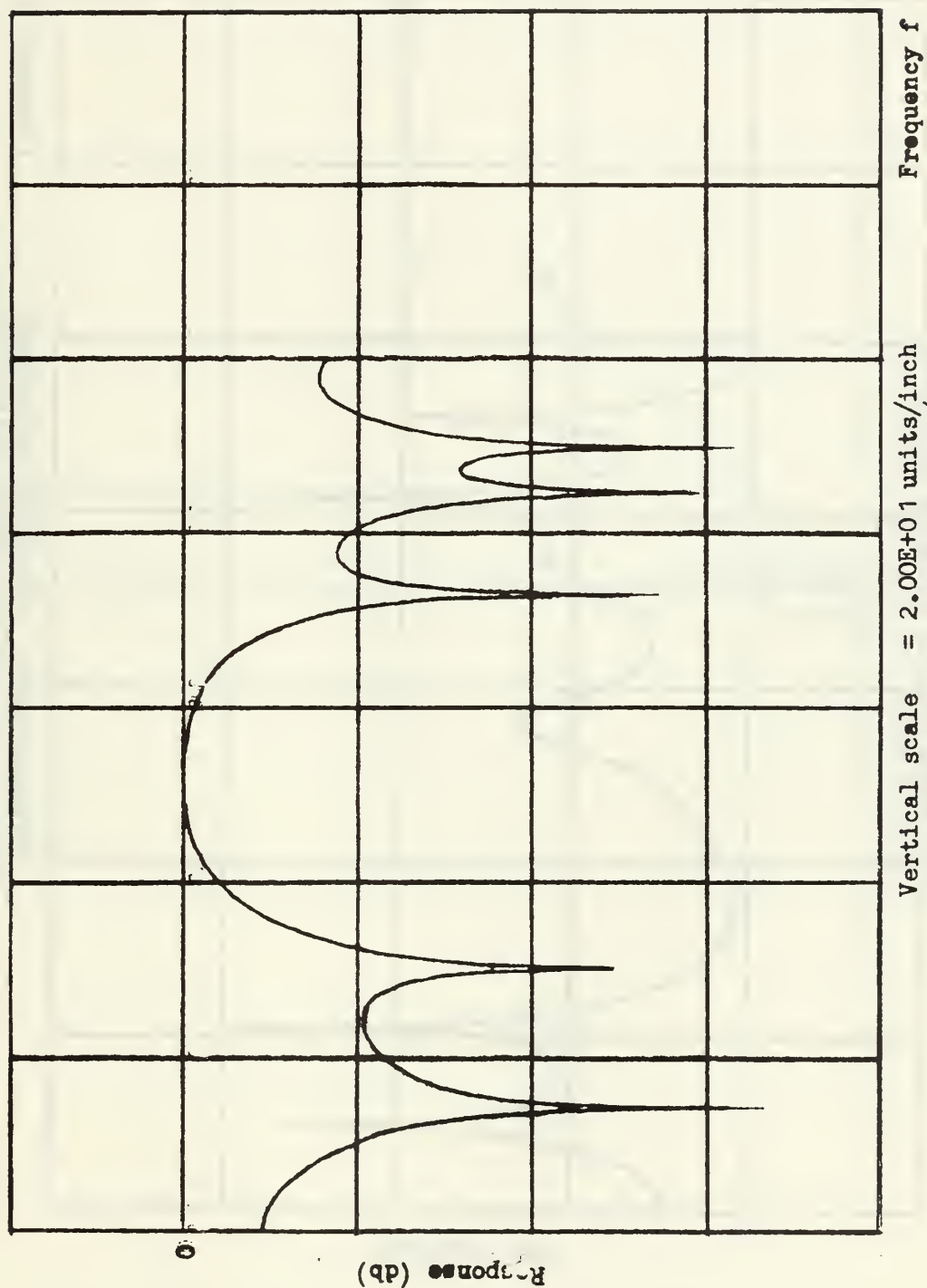
Vertical scale = $2.00E+01$ units/inch
 Horizontal scale = $2.00E+02$ units/inch
 Frequency responses for $\theta = 0^\circ$ after adapting
 for 300 cycles of the desired frequency 500 Hz.



Vertical scale = $2.00E+01$ units/inch

Horizontal scale = $2.00E+02$ units/inch

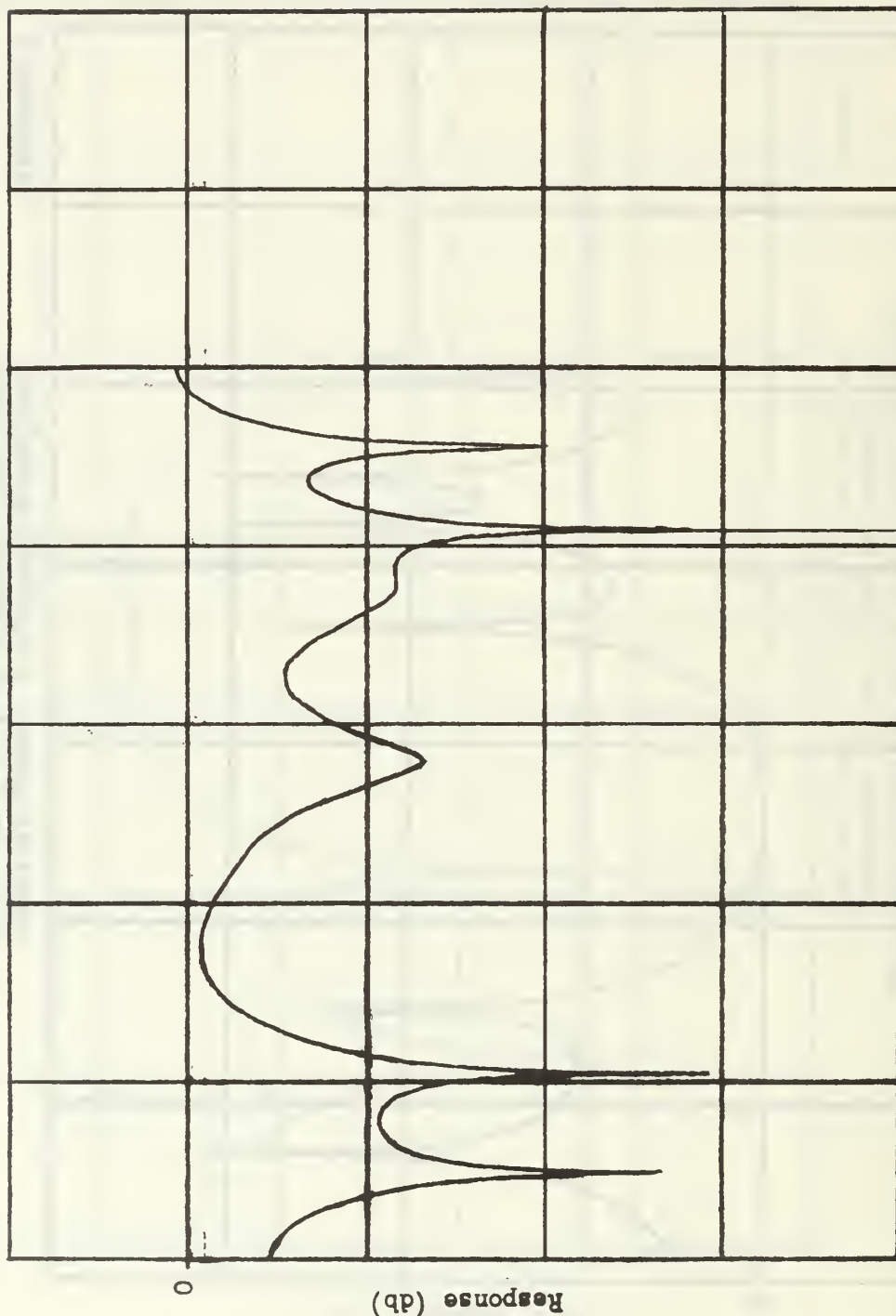
Frequency responses for $\theta = 30^\circ$ after adapting for 300 cycles of the desired frequency 500 Hz.



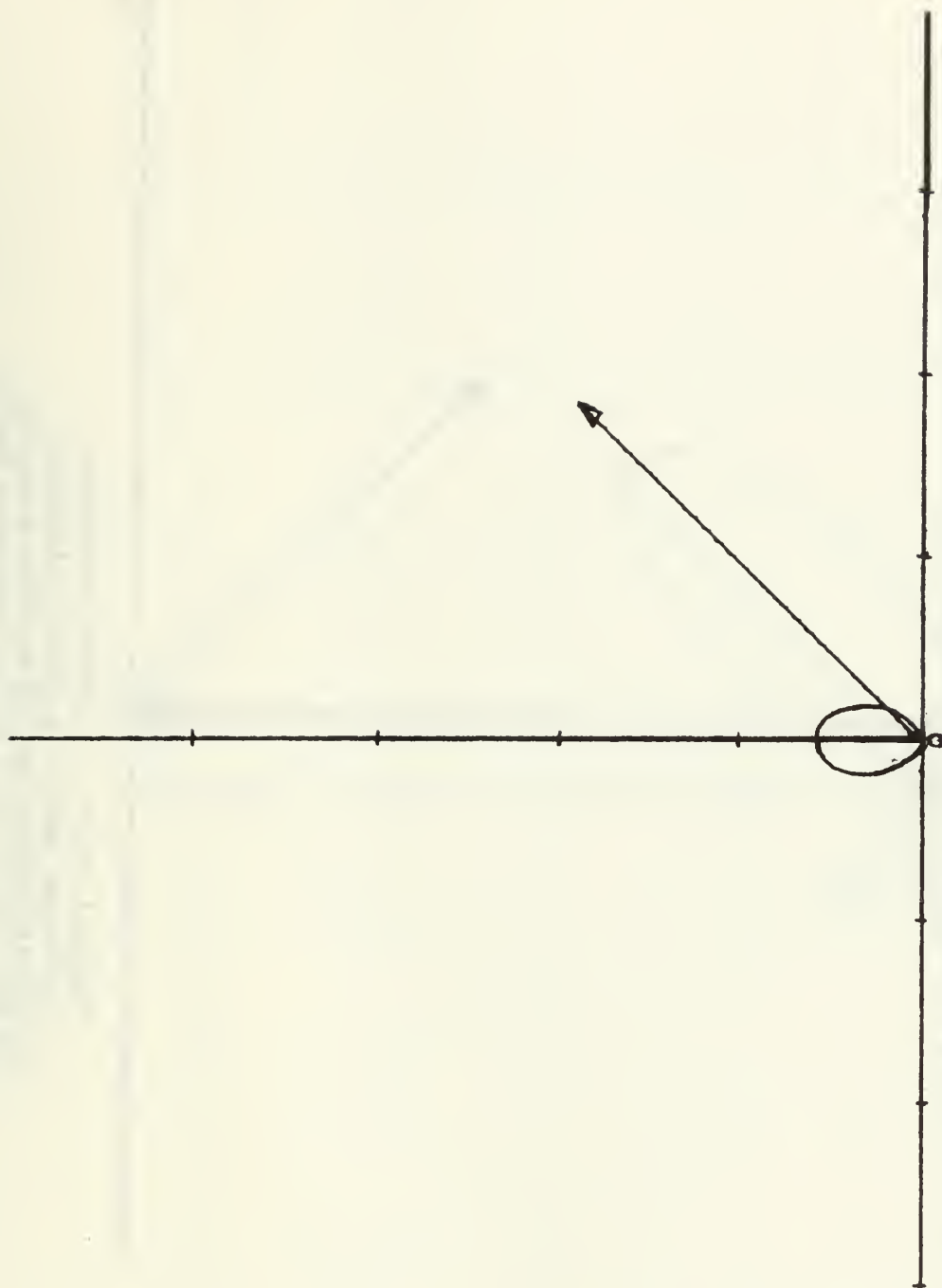
Vertical scale = 2.00E+01 units/inch

Horizontal scale = 2.00E+02 units/inch

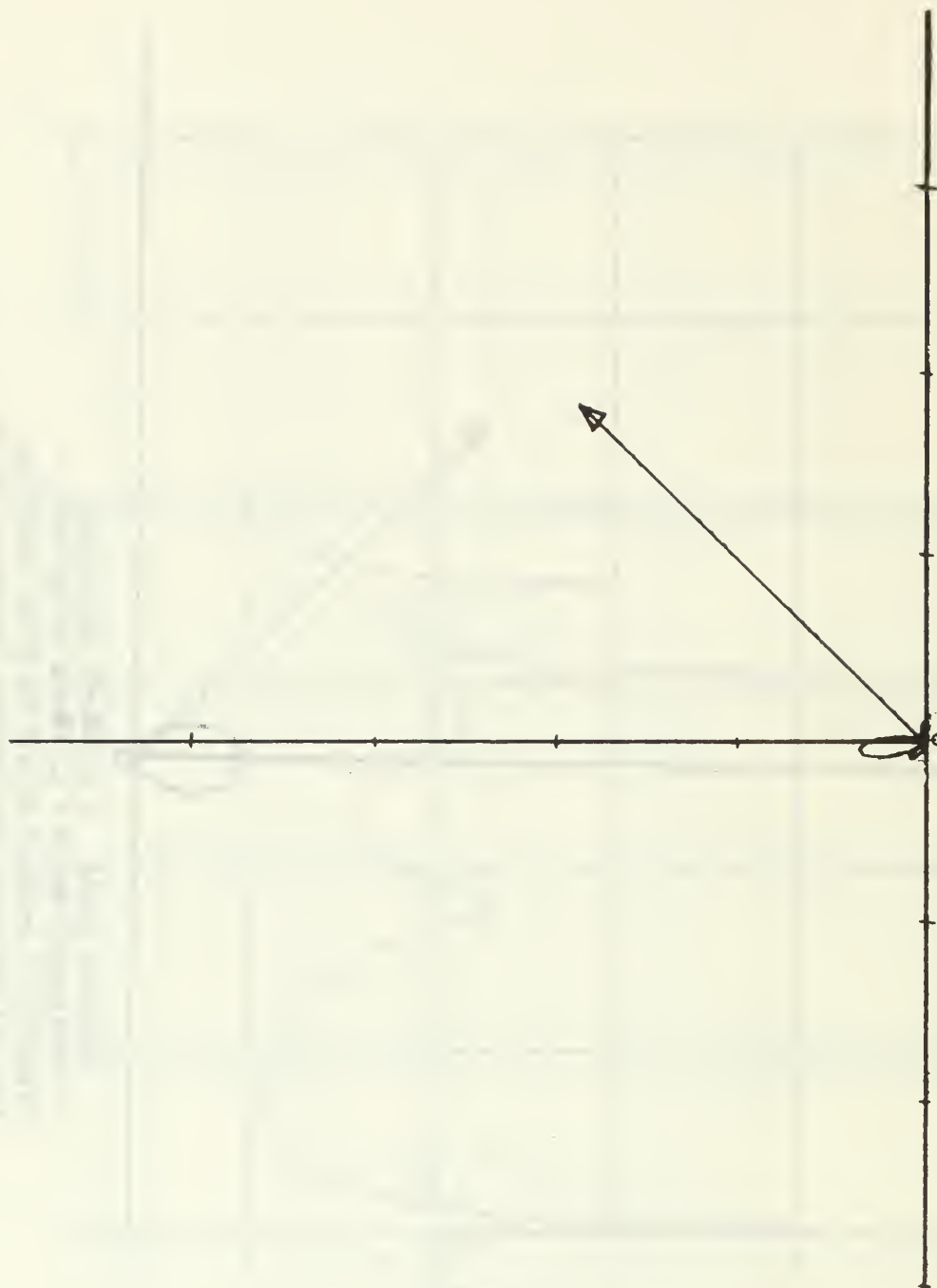
Frequency responses for $\theta = 45^\circ$ after adapting for 300 cycles of the desired frequency 500 Hz.



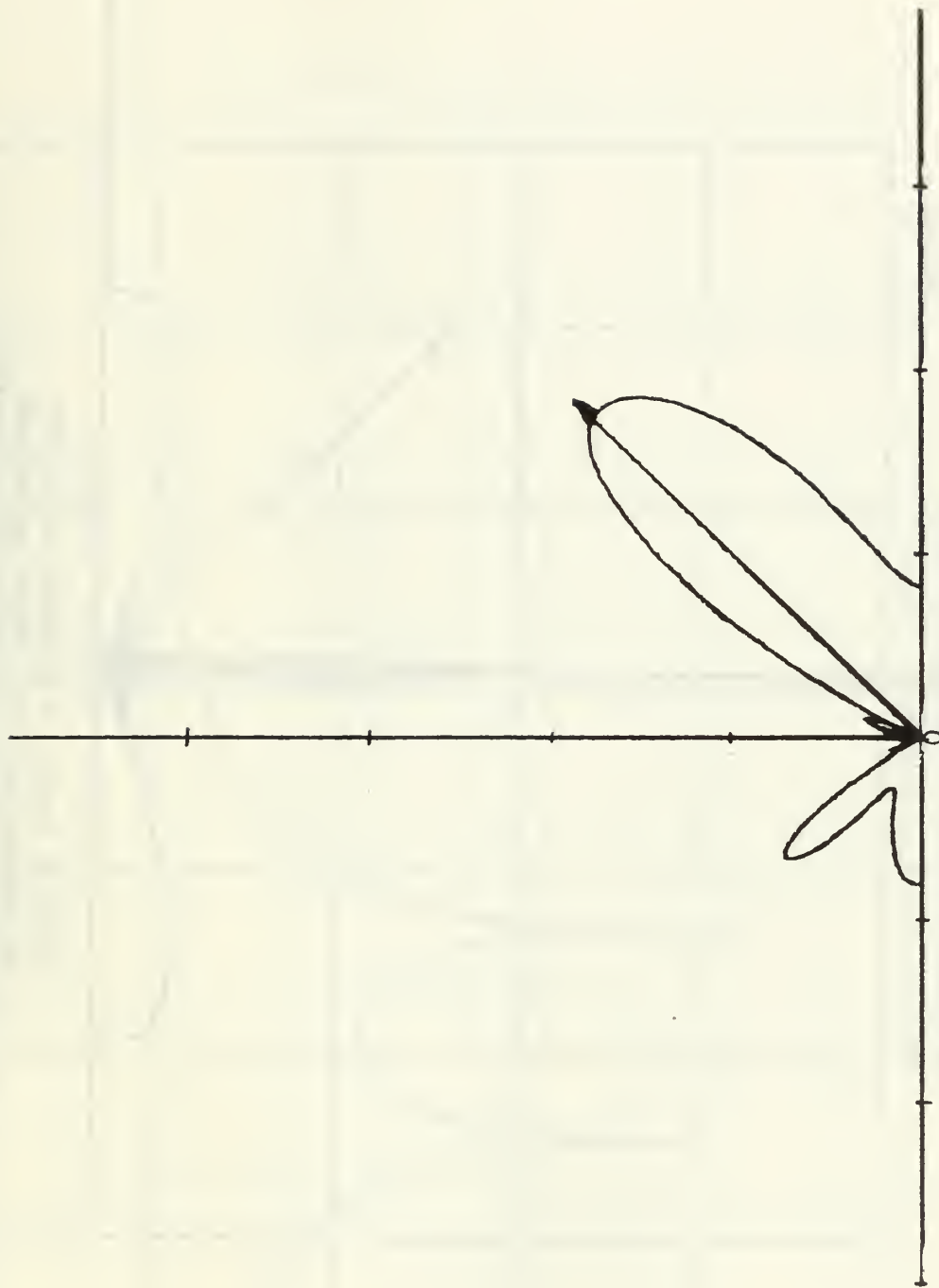
Vertical scale = $2.00E+01$ units/inch
 Horizontal scale = $2.00E+02$ units/inch
 Frequency responses for $\theta = 88.80$ after adapting
 for 300 cycles of the desired frequency 500 Hz.



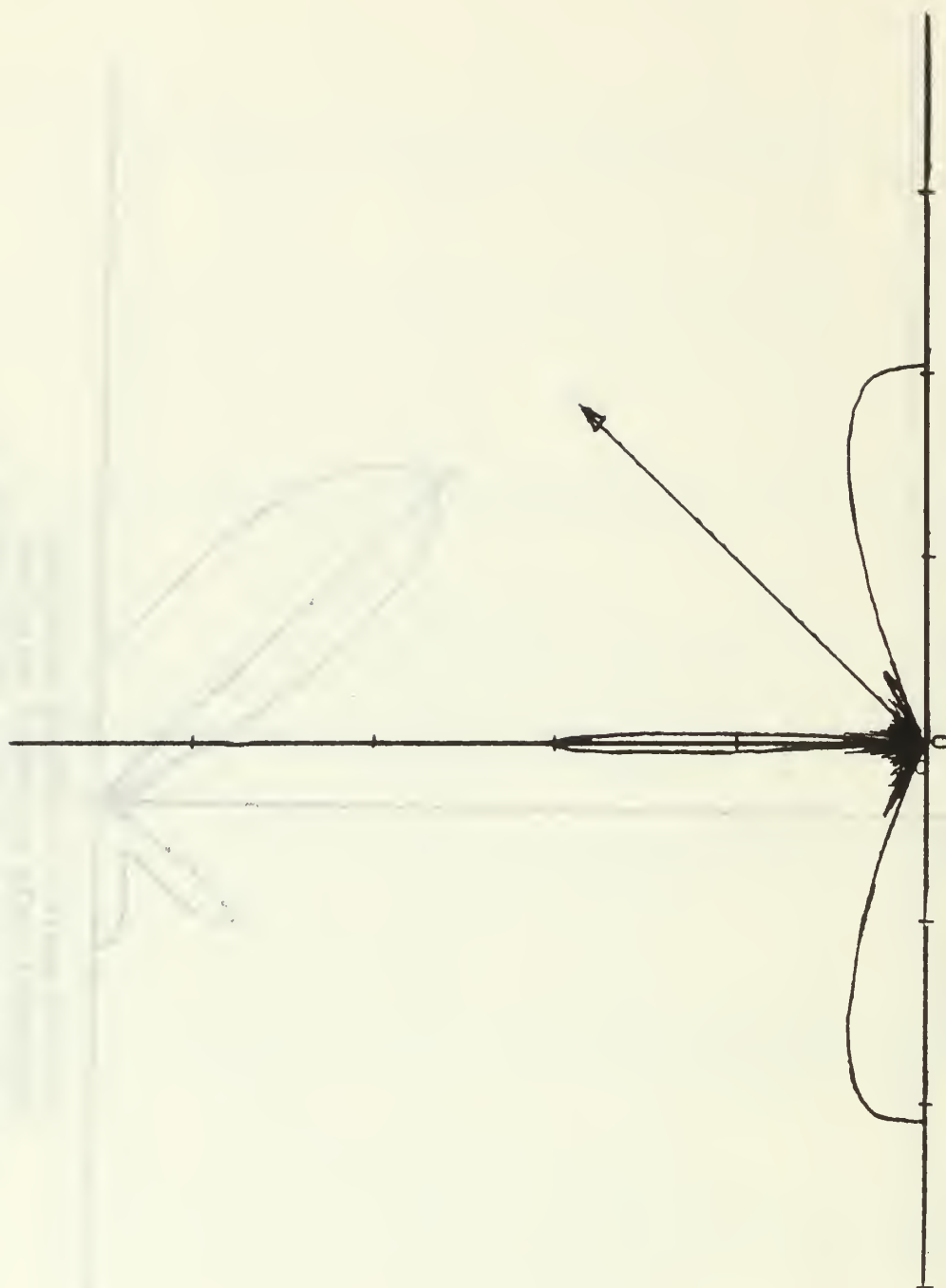
Vertical scale = 4.00E-01 units/inch
Horizontal scale = 4.00E-01 units/inch
Directivity pattern at 100 Hz after adapting
for 600 cycles of the desired frequency 500 Hz.



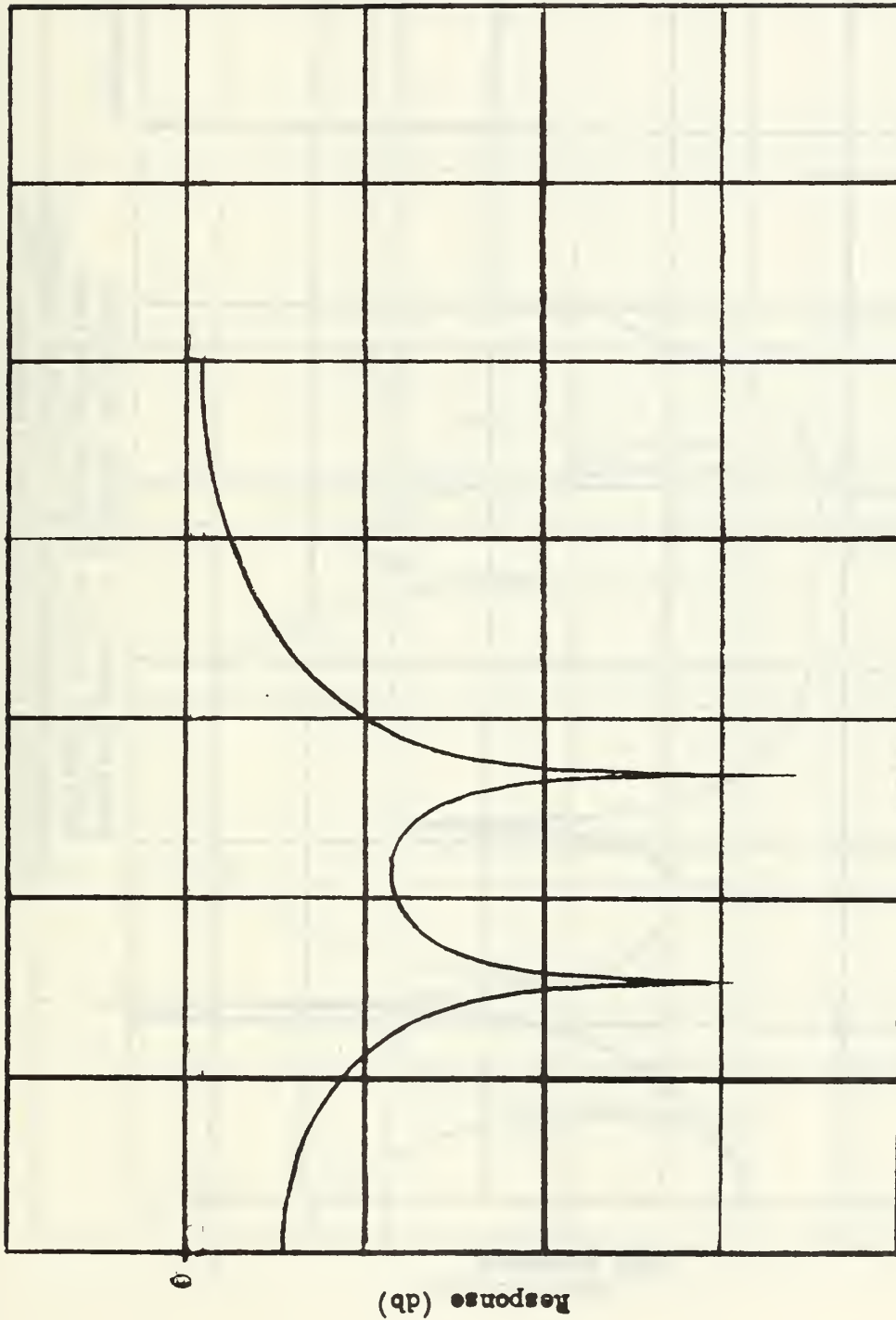
Vertical scale = 4.00E-01 units/inch
Horizontal scale = 4.00E-01 units/inch
Directivity pattern at 200 Hz after adapting
for 600 cycles of the desired frequency 500 Hz.



Vertical scale = 4.00E-01 units/inch
Horizontal scale = 4.00E-01 units/inch
Directivity pattern at 500 Hz after adapting
for 600 cycles of the desired frequency 500 Hz.



Vertical scale = 4.00E-01 units/inch
Horizontal scale = 4.00E-01 units/inch
Directivity pattern at 1000 Hz after adapting
for 600 cycles of the desired frequency 500 Hz.

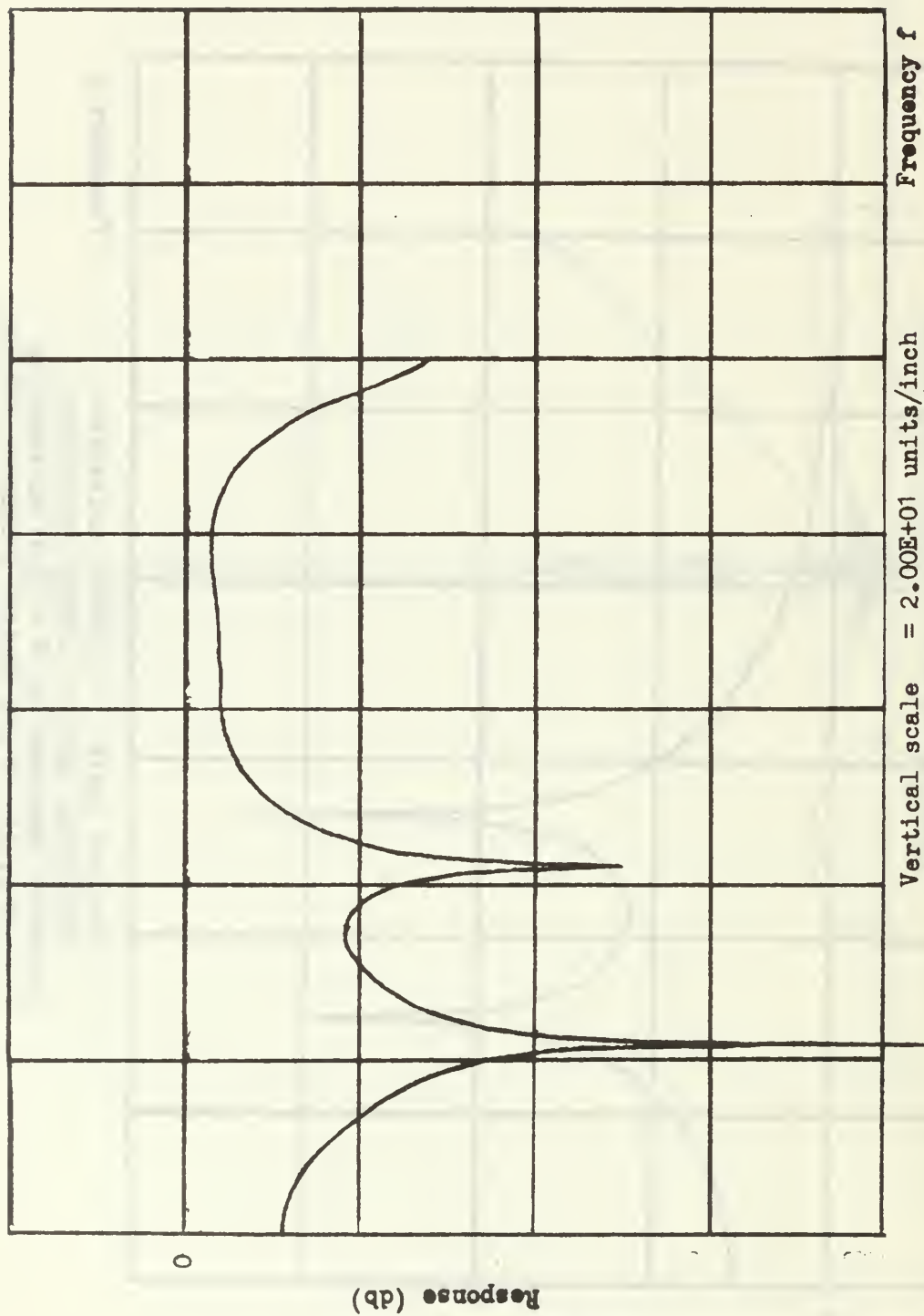


Frequency f

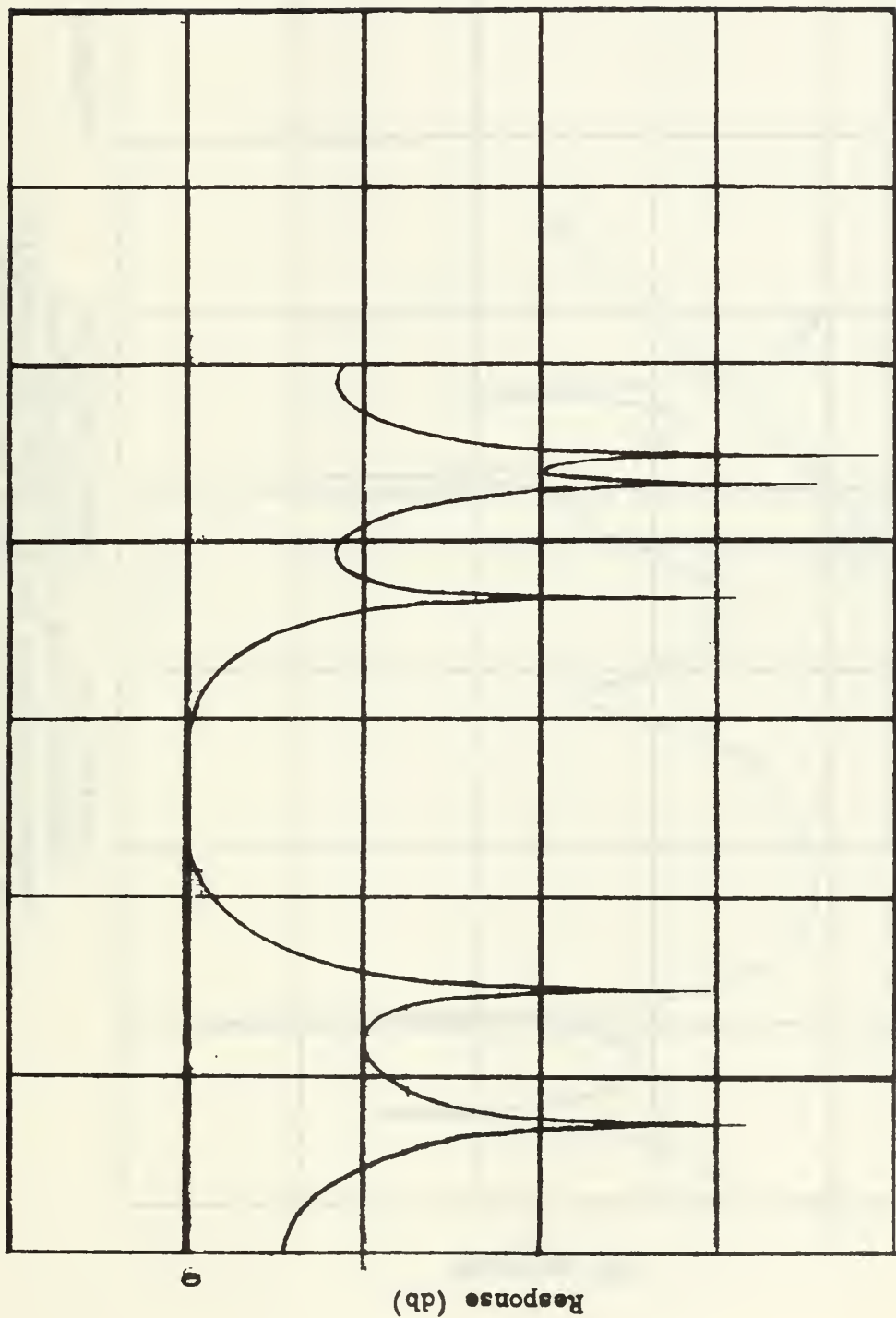
Vertical scale = $2.00E+01$ units/inch

Horizontal scale = $2.00E+02$ units/inch

Frequency responses for $\theta = 0^\circ$ after adapting for 600 cycles of the desired frequency 500 Hz.



Vertical scale = 2.00E+01 units/inch
 Horizontal scale = 2.00E+02 units/inch
 Frequency responses for $\theta = 30^\circ$ after adapting
 for 600 cycles of the desired frequency 500 Hz.

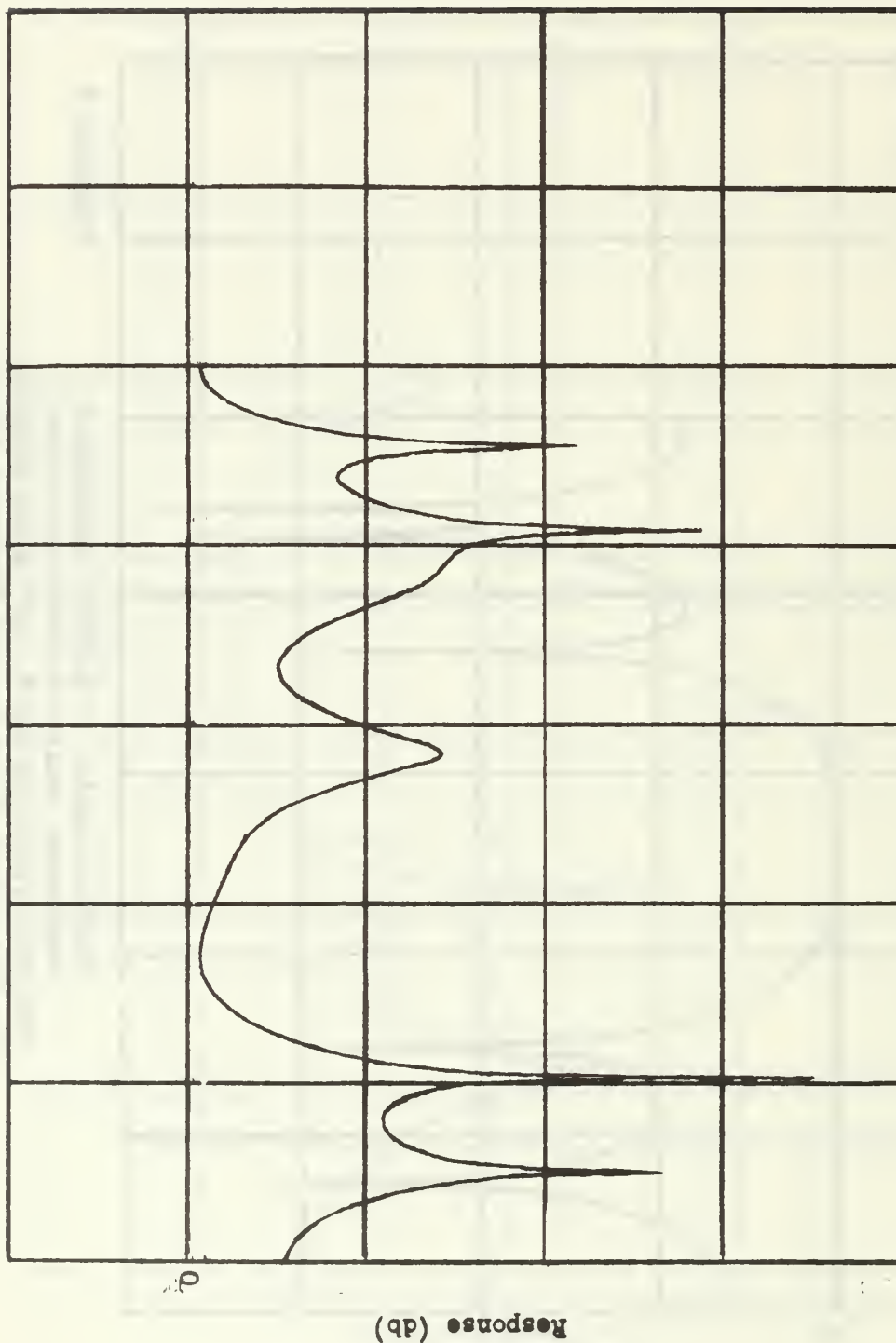


Frequency f

Vertical scale = 2.00E+01 units/inch

Horizontal scale = 2.00E+02 units/inch

Frequency responses for $\theta = 45^\circ$ after adapting for 600 cycles of the desired frequency 500 Hz.

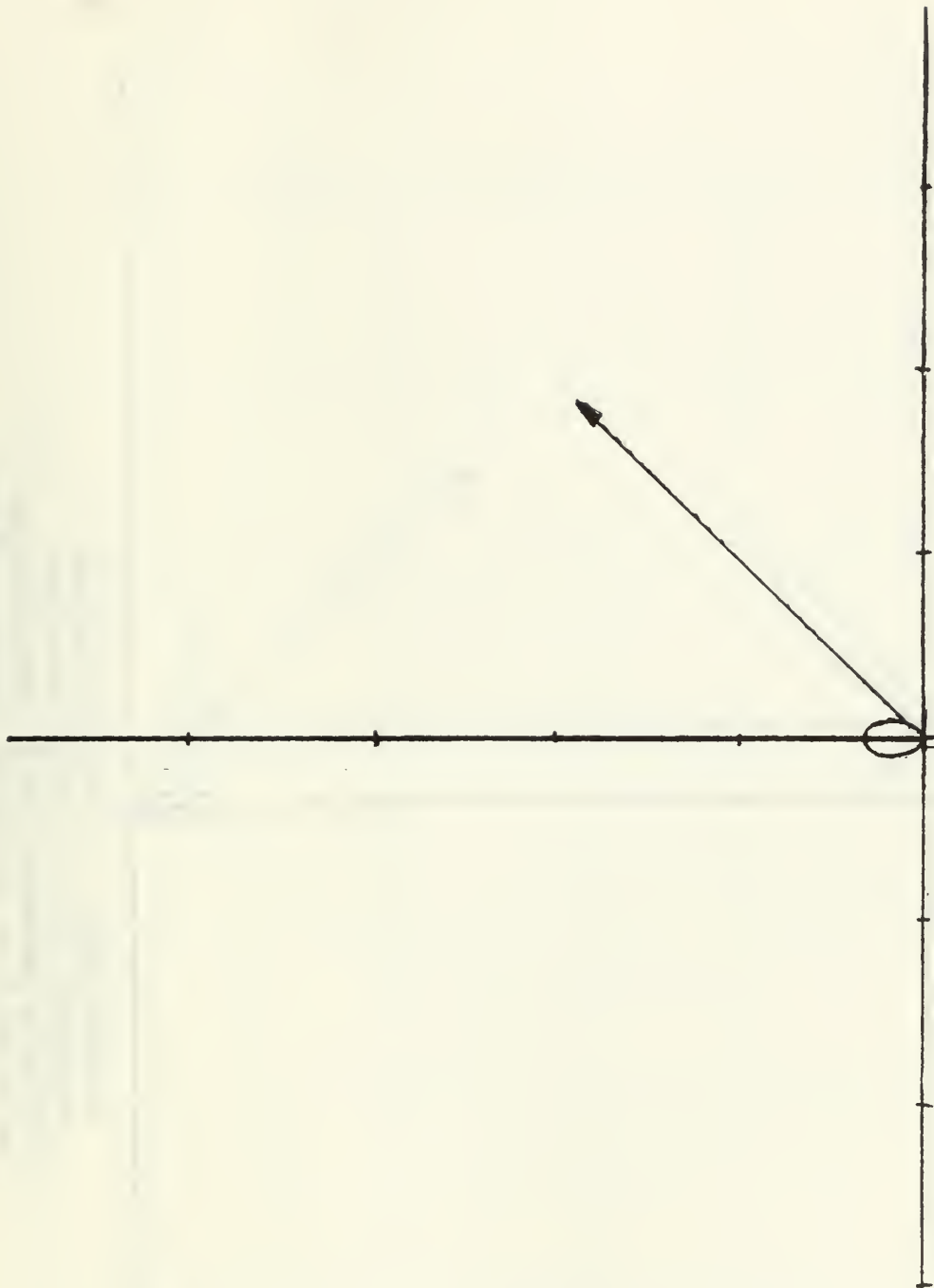


Frequency f

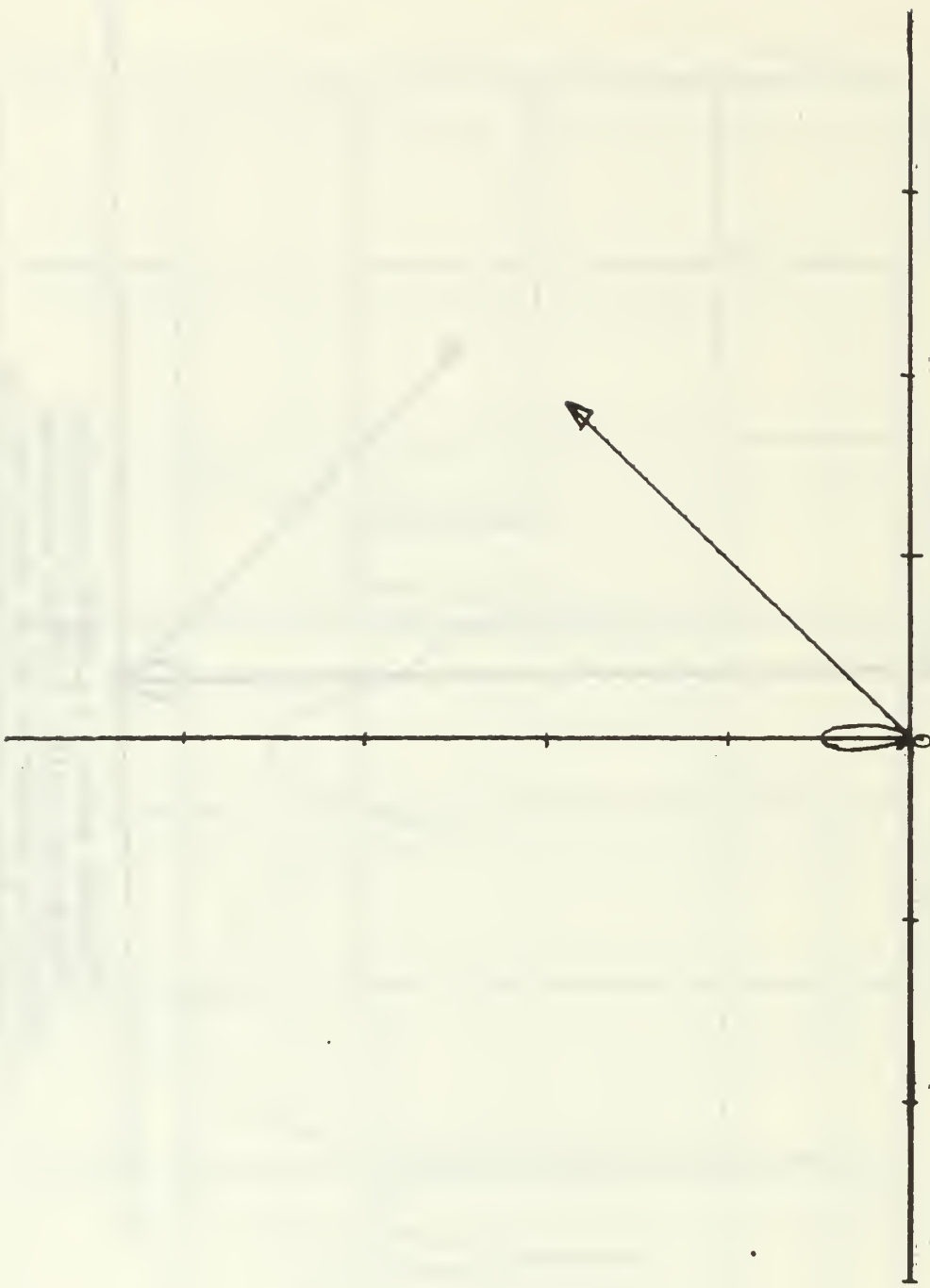
Vertical scale = $2.00E+01$ units/inch

Horizontal scale = $2.00E+02$ units/inch

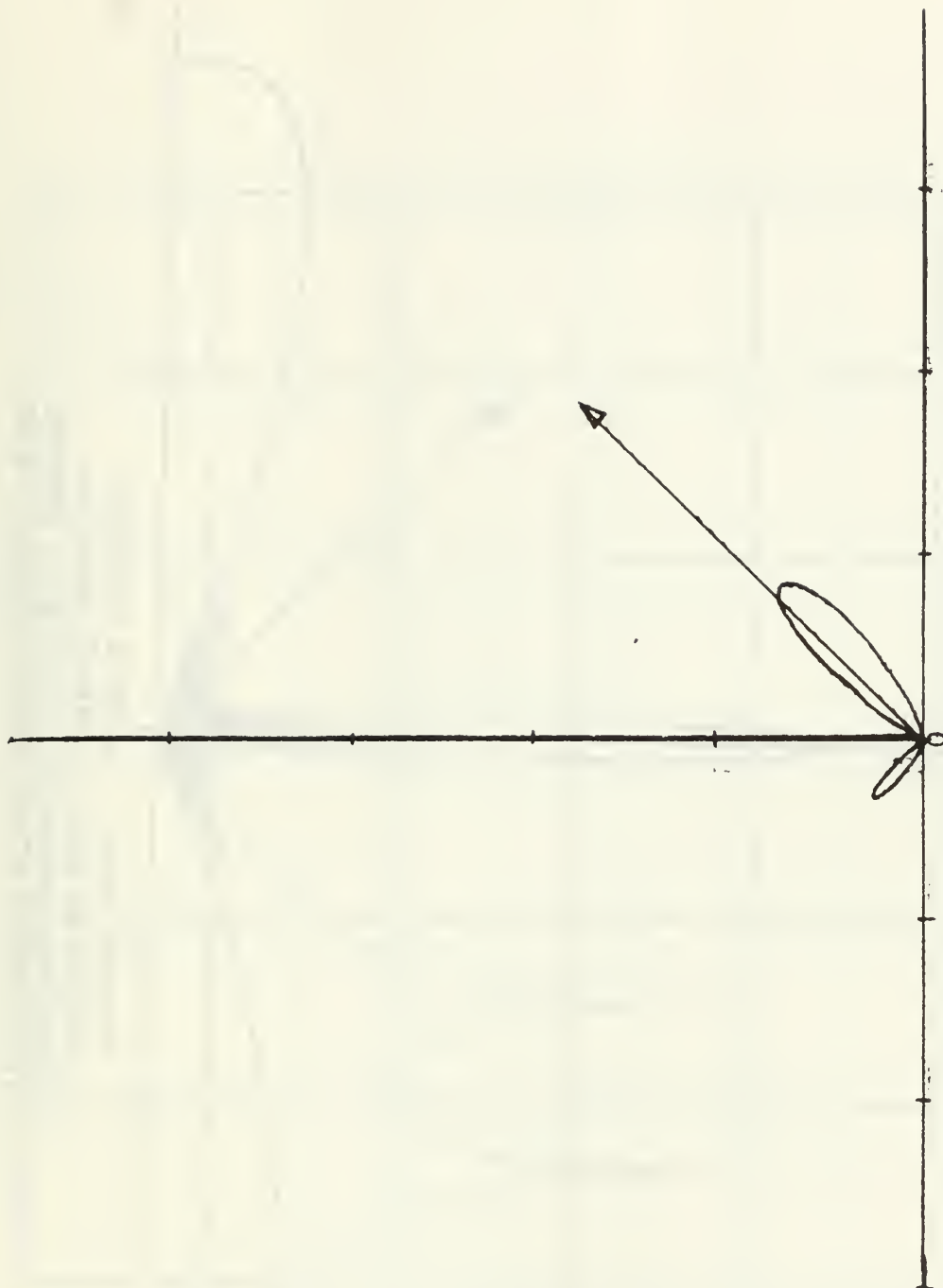
Frequency responses for $\theta = 88.8^\circ$ after adapting for 600 cycles of the desired frequency 500 Hz.



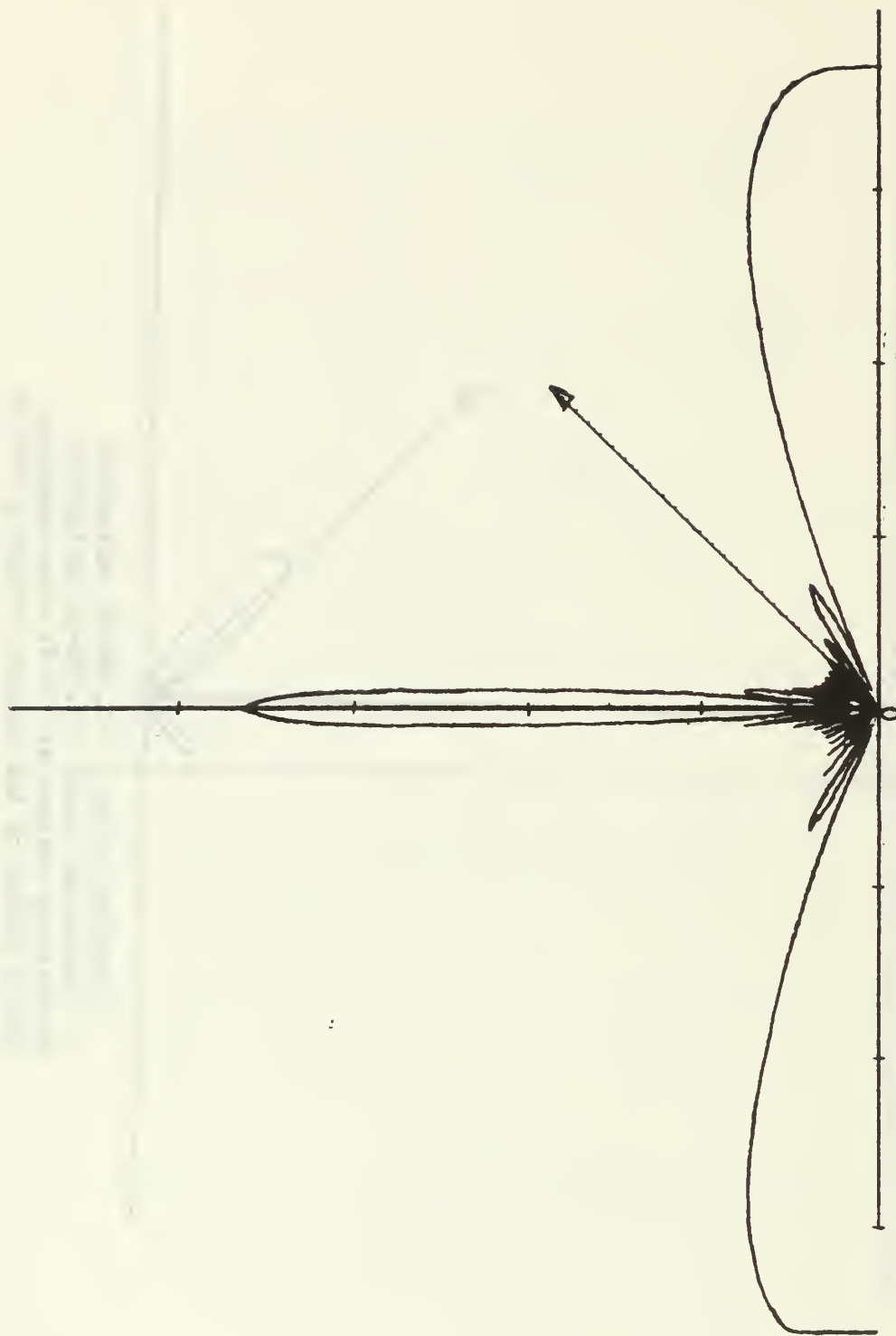
Vertical scale = 4.00E-01 units/inch
Horizontal scale = 4.00E-01 units/inch
Directivity pattern at 100 Hz after adapting
for 15 cycles of the desired frequency 500 Hz.



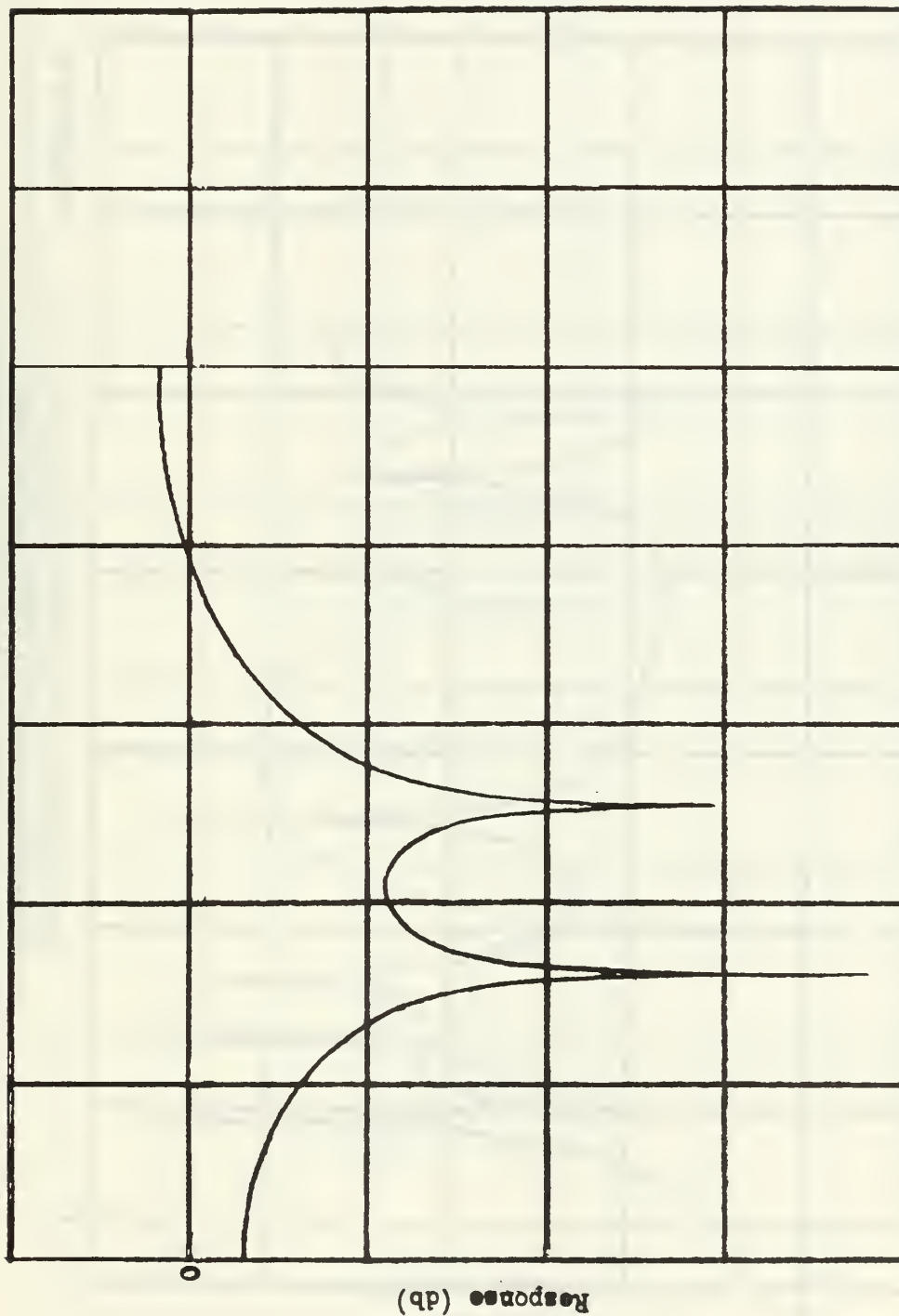
Vertical scale = 4.00E-01 units/inch
Horizontal scale = 4.00E-01 units/inch
Directivity pattern at 200 Hz after adapting
for 15 cycles of the desired frequency 500 Hz.



Vertical scale = 4.00E-01 units/inch
Horizontal scale = 4.00E-01 units/inch
Directivity pattern at 500 Hz after adapting
for 15 cycles of the desired frequency 500 Hz.



Vertical scale = $4.000-01$ units/inch
Horizontal scale = $4.000E-01$ units/inch
Directivity pattern at 1000 Hz after adapting
for 15 cycles of the desired frequency 500 Hz.

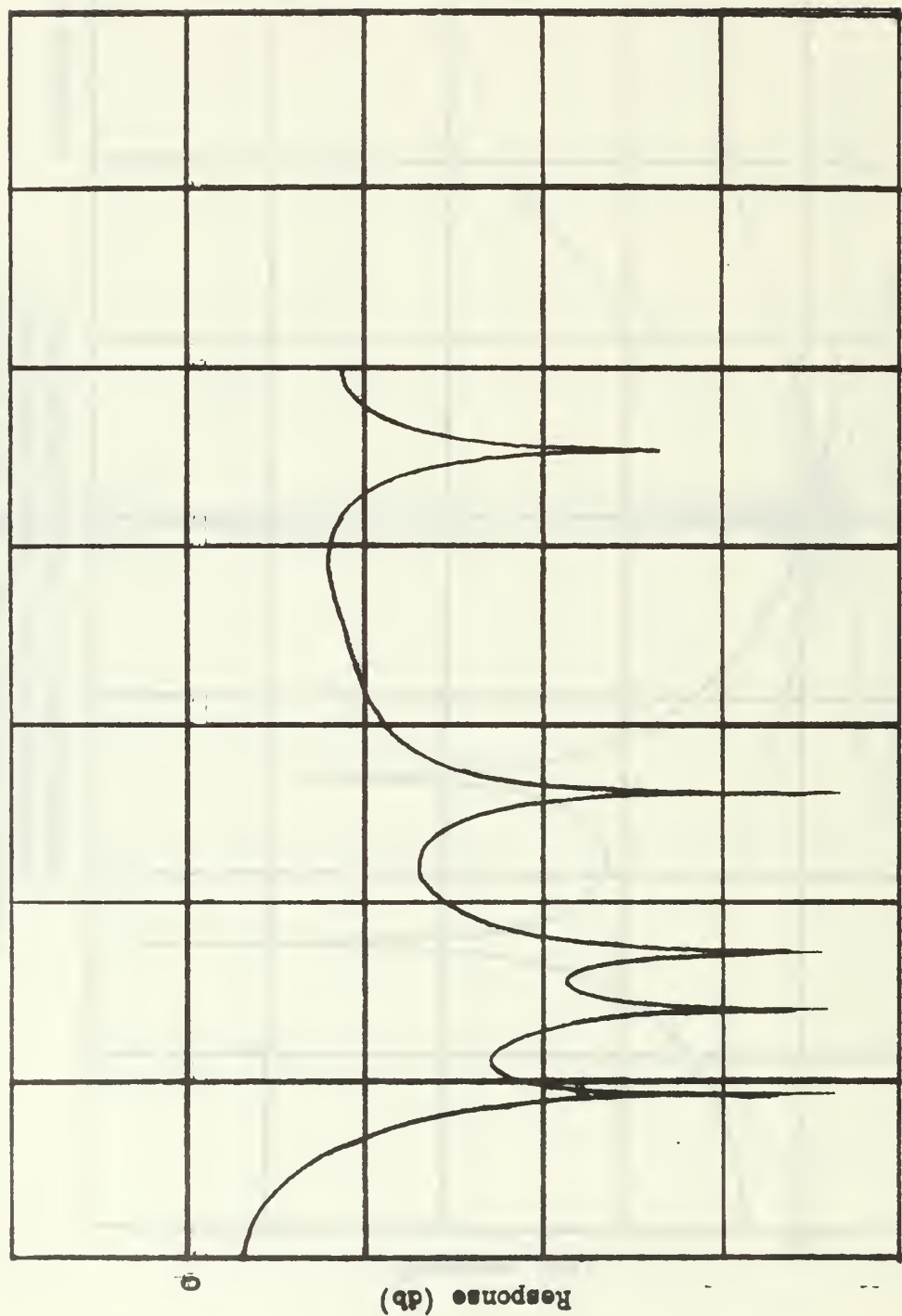


Frequency f

Vertical scale = 2.00E+01 units/inch

Horizontal scale = 2.00E+02 units/inch

Frequency responses for $\theta = 0^\circ$ after adapting for 15 cycles of the desired frequency 500 Hz.

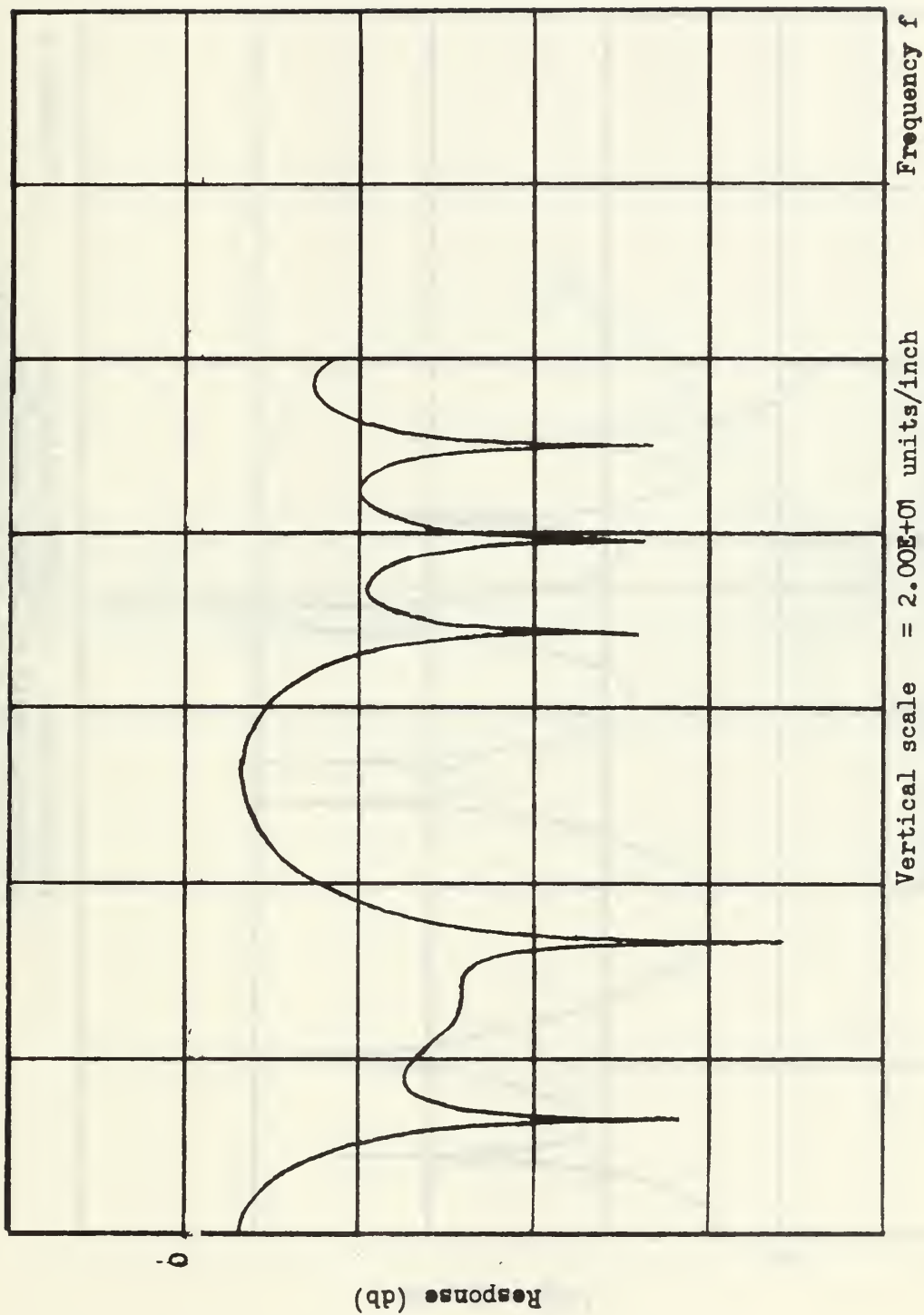


Frequency f

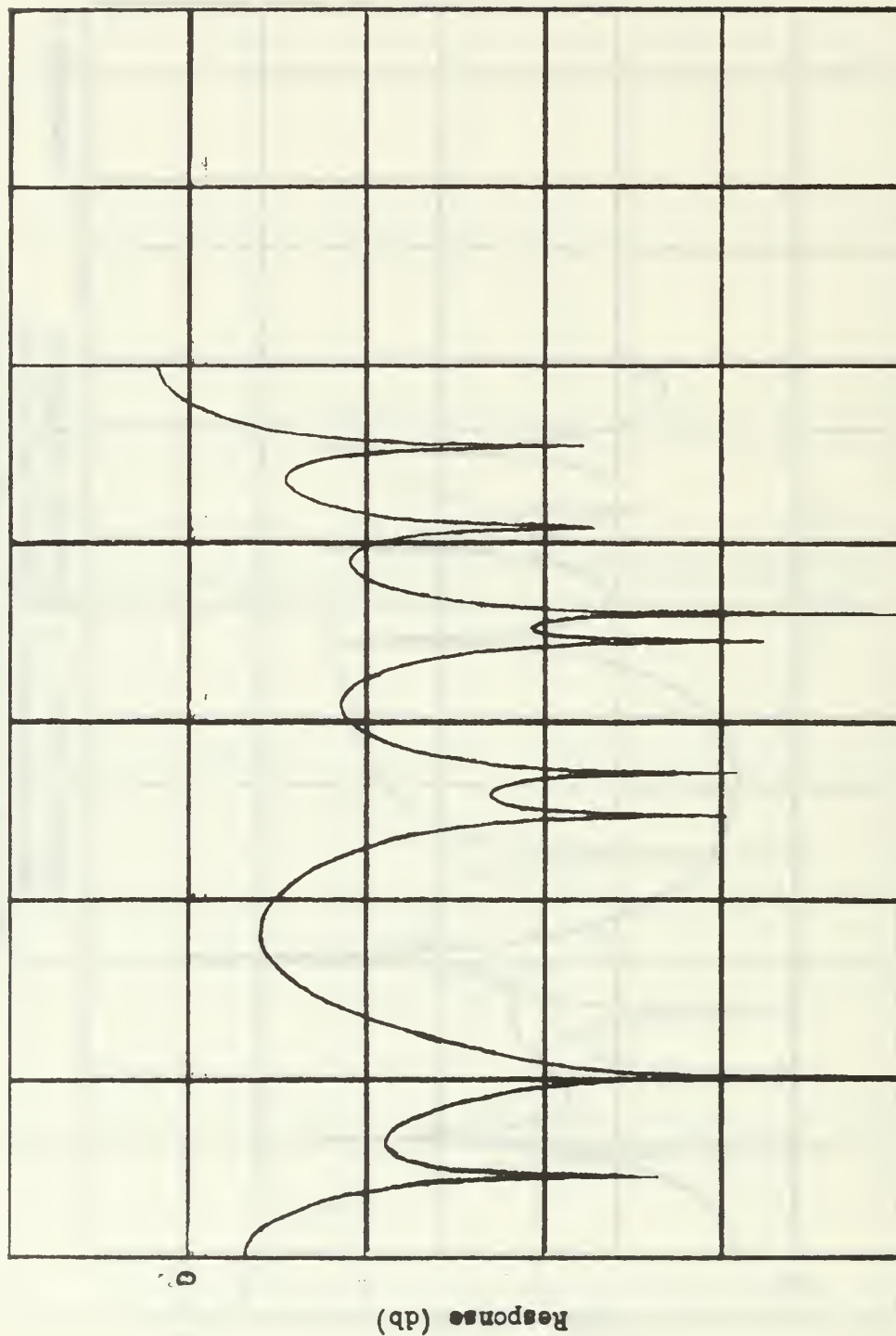
Vertical scale = 2.00E+01 units/inch

Horizontal scale = 2.00E+02 units/inch

Frequency responses for $\theta = 30^\circ$ after adapting for 15 cycles of the desired frequency 500 Hz.



Vertical scale = 2.00E+01 units/inch
 Horizontal scale = 2.00E+02 units/inch
 Frequency responses for $\theta = 45^\circ$ after adapting
 for 15 cycles of the desired frequency 500 Hz.

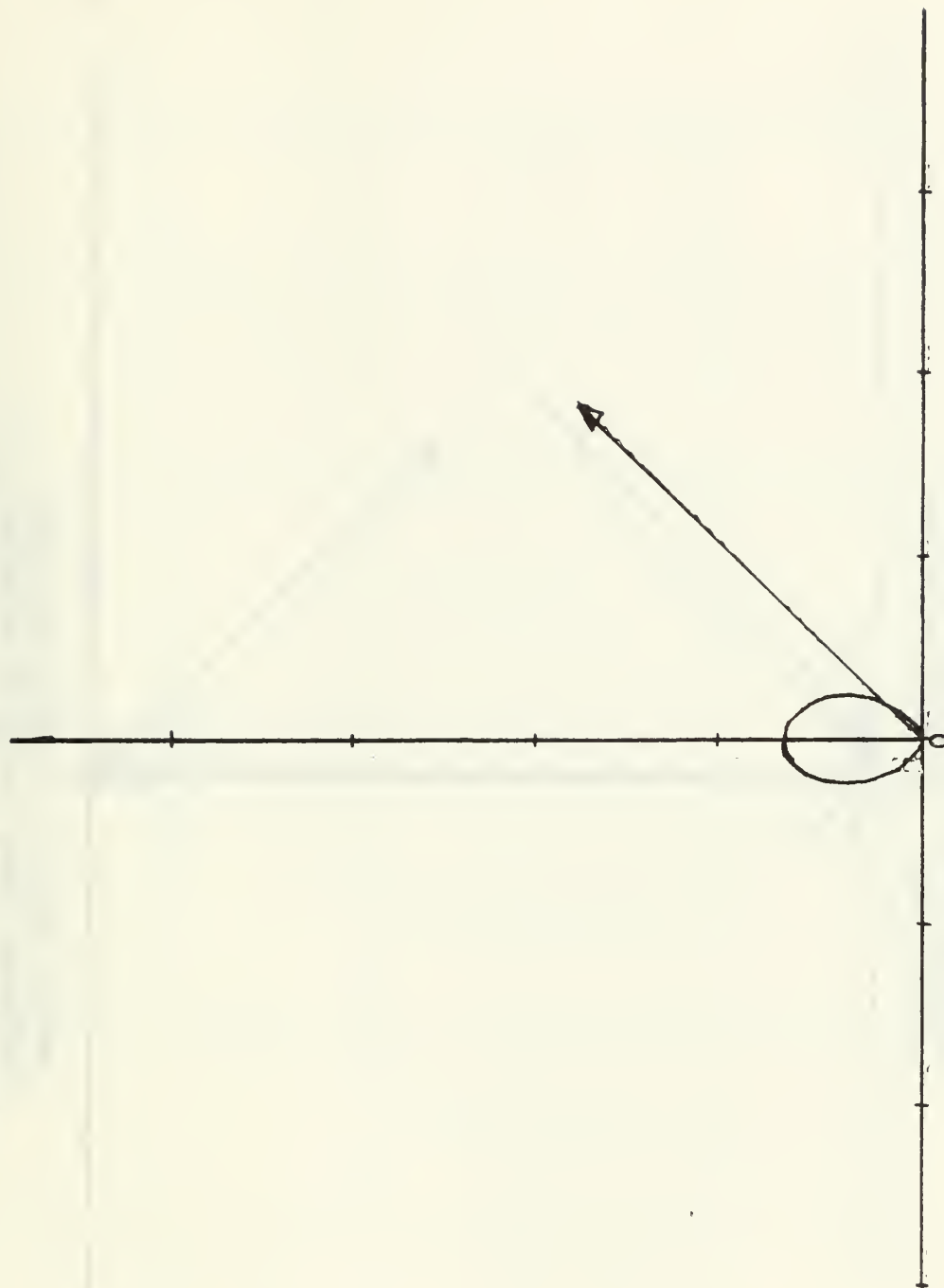


Frequency f

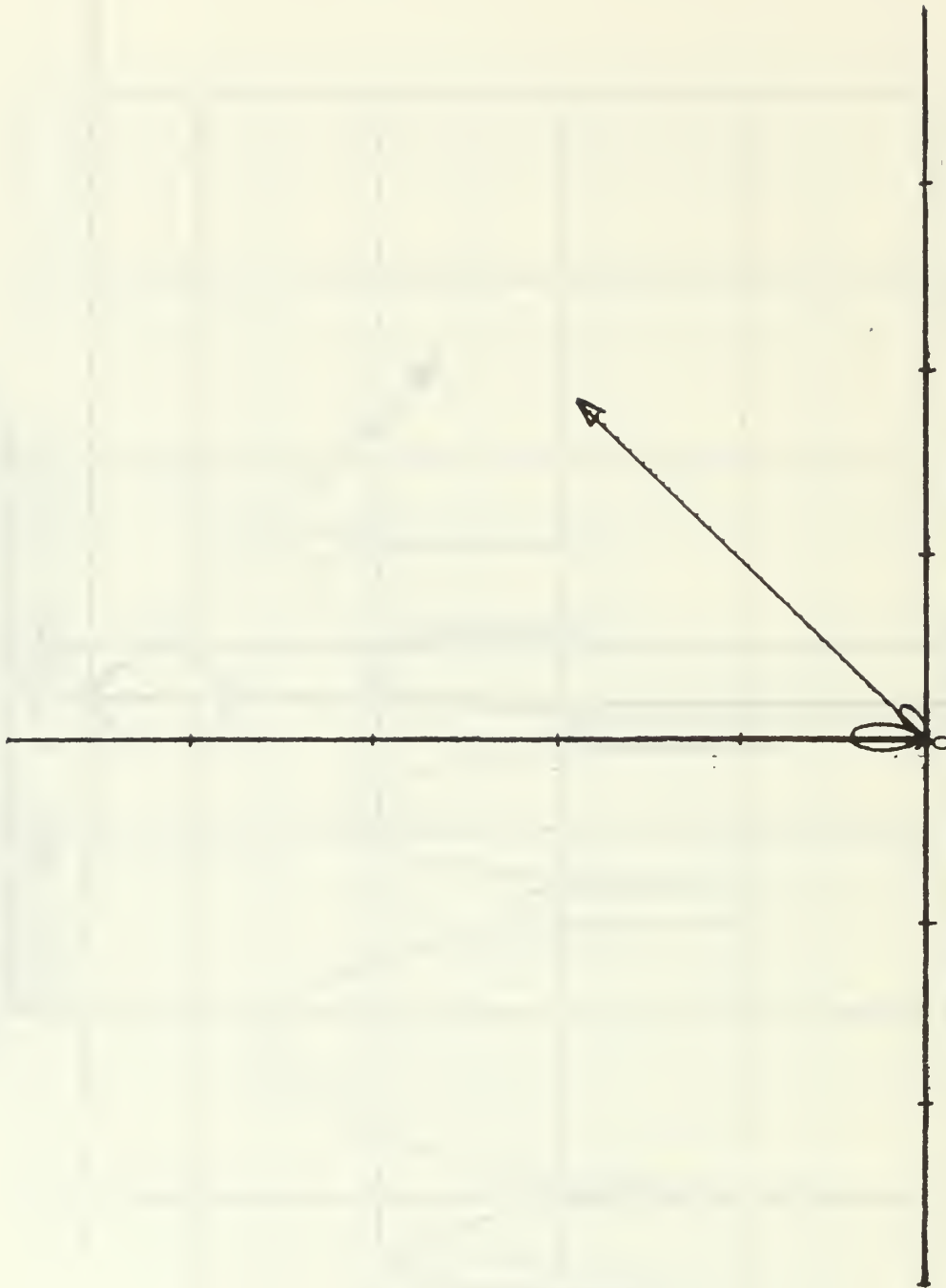
Vertical scale = $2.00E+01$ units/inch

Horizontal scale = $2.00E+02$ units/inch

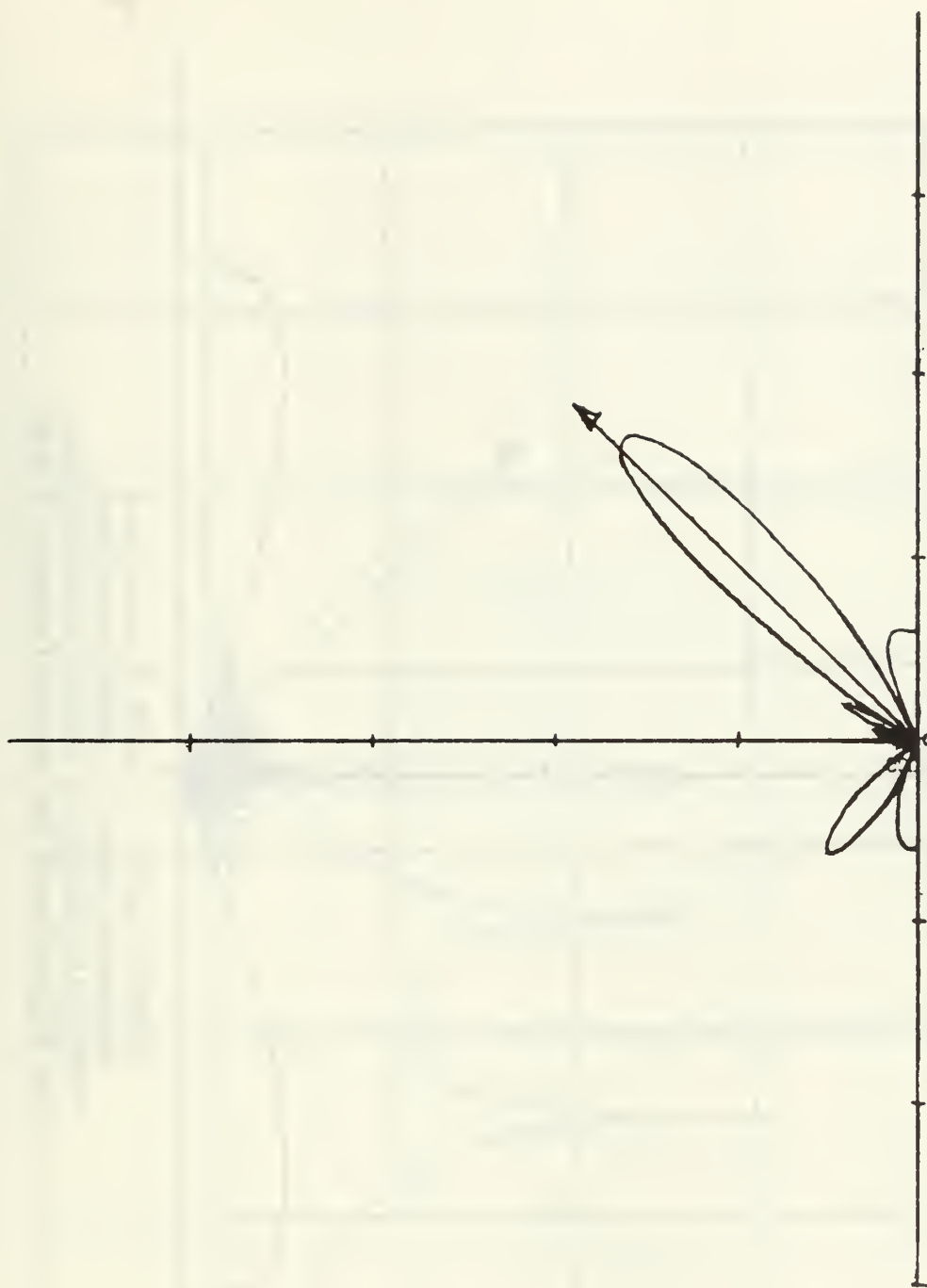
Frequency responses for $\theta = 88.8^\circ$ after adapting for 15 cycles of the desired frequency 500 Hz.



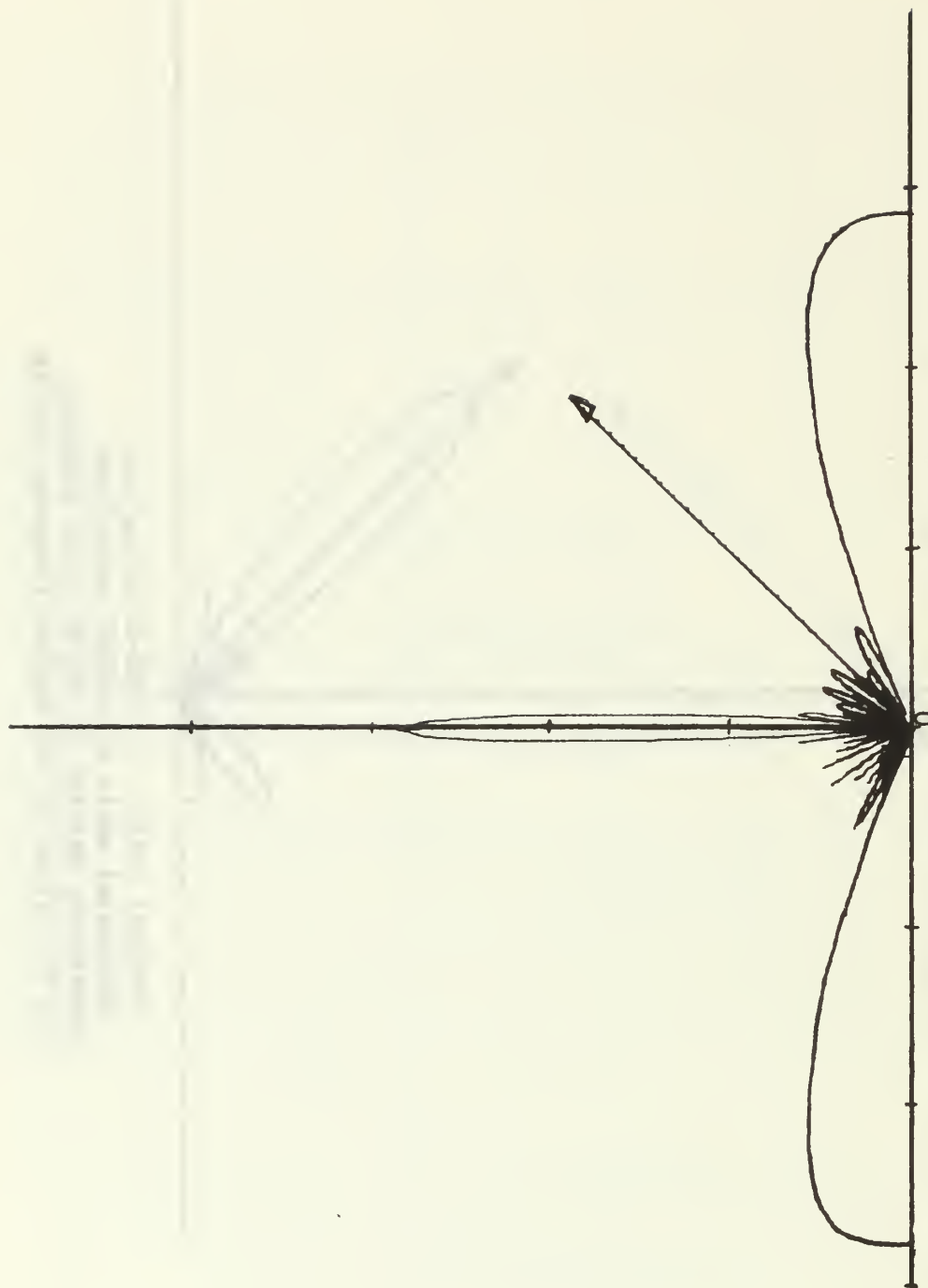
Vertical scale = 4.00E-01 units/inch
Horizontal scale = 4.00E-01 units/inch
Directivity pattern at 100 Hz after adapting
for 300 cycles of the desired frequency 500 Hz.



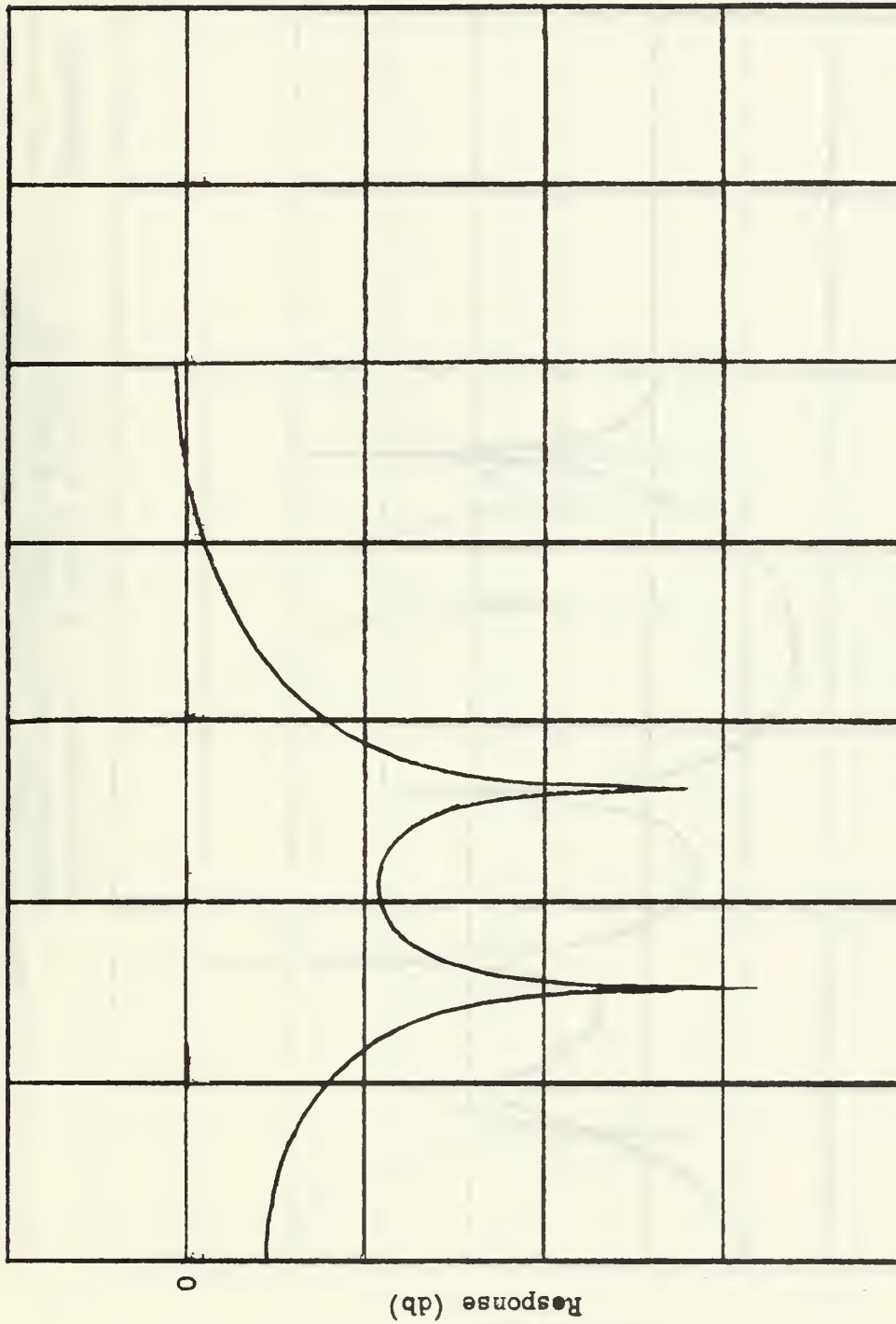
Vertical scale = 4.00E-01 units/inch
Horizontal scale = 4.00E-01 units/inch
Directivity pattern at 200 Hz after adapting
for 300 cycles of the desired frequency 500 Hz.



Vertical scale = 4.00E-01 units/inch
Horizontal scale = 4.00E-01 units/inch
Directivity pattern at 500 Hz after adapting
for 300 cycles of the desired frequency 500 Hz.



Vertical scale = 4.00E-01 units/inch
Horizontal scale = 4.00E-01 units/inch
Directivity pattern at 1000 Hz after adapting
for 300 cycles of the desired frequency 500 Hz.

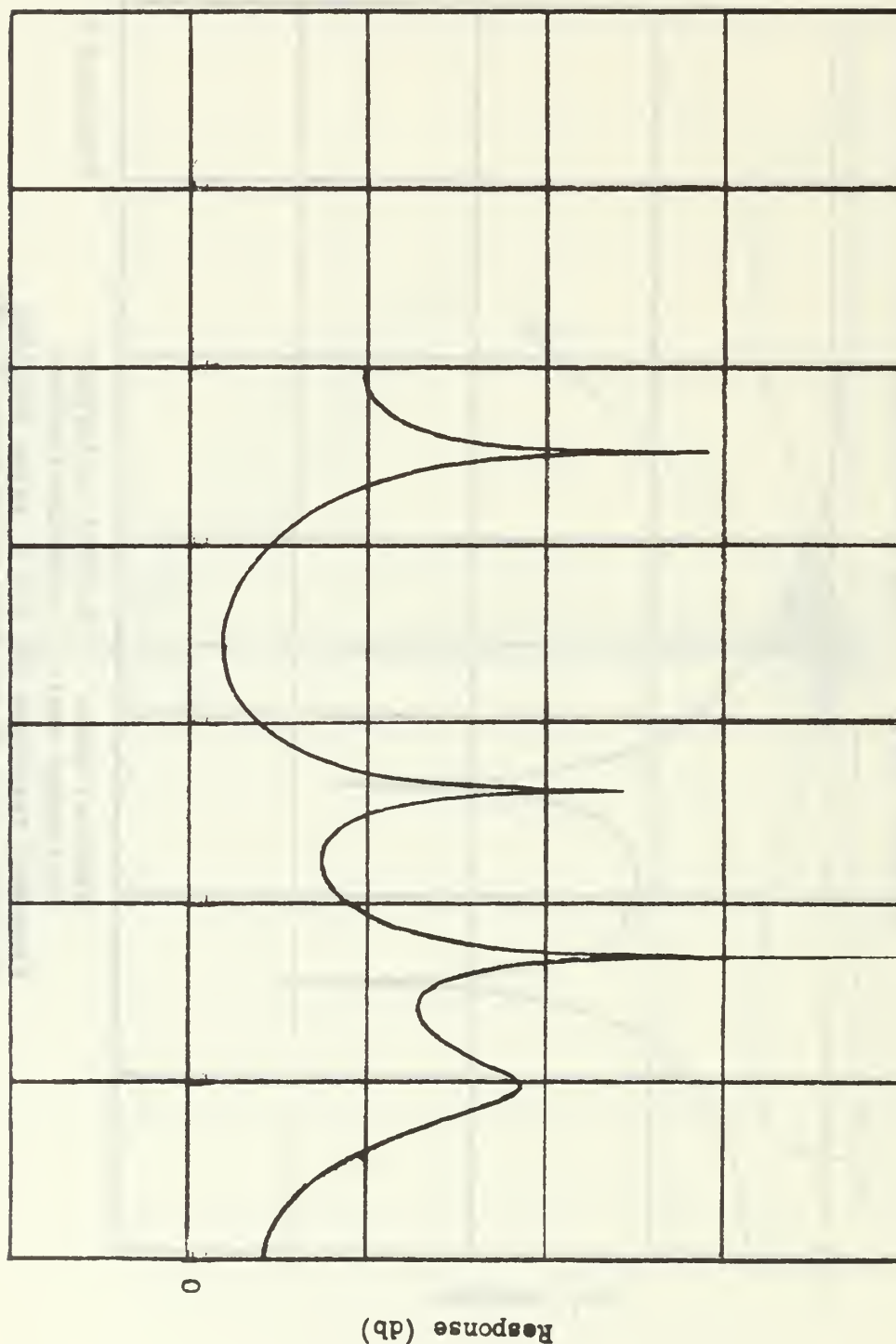


Frequency f

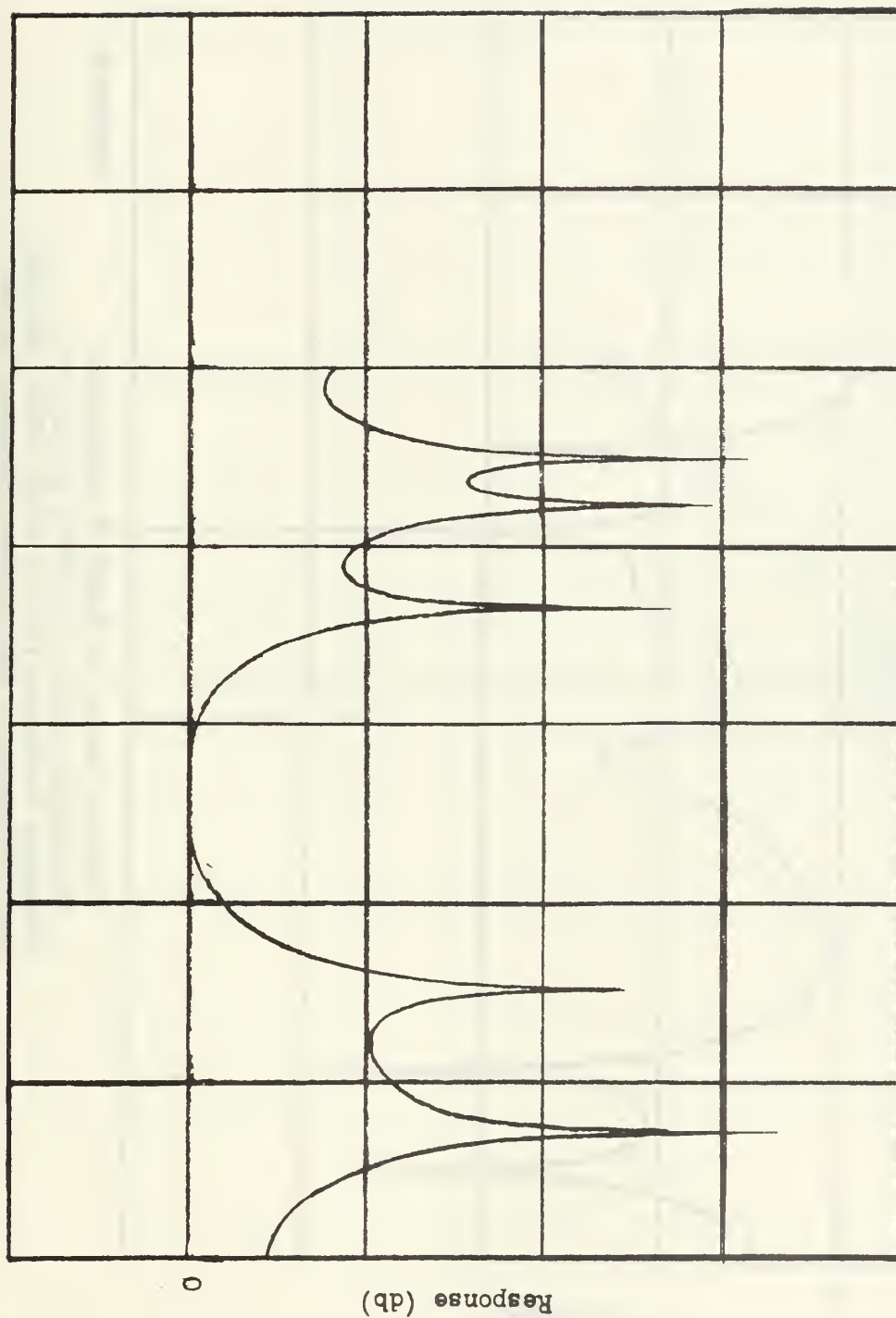
Vertical scale = 2.00 units/inch

Horizontal scale = 2.00 units/inch

Frequency responses for $\theta = 0^\circ$ after adapting for 300 cycles of the desired frequency 500 Hz.



Vertical scale = $2.00E+01$ units/inch
 Horizontal scale = $2.00E+02$ units/inch
 Frequency responses for $\theta = 300$ after adapting
 for 300 cycles of the desired frequency 500 Hz.

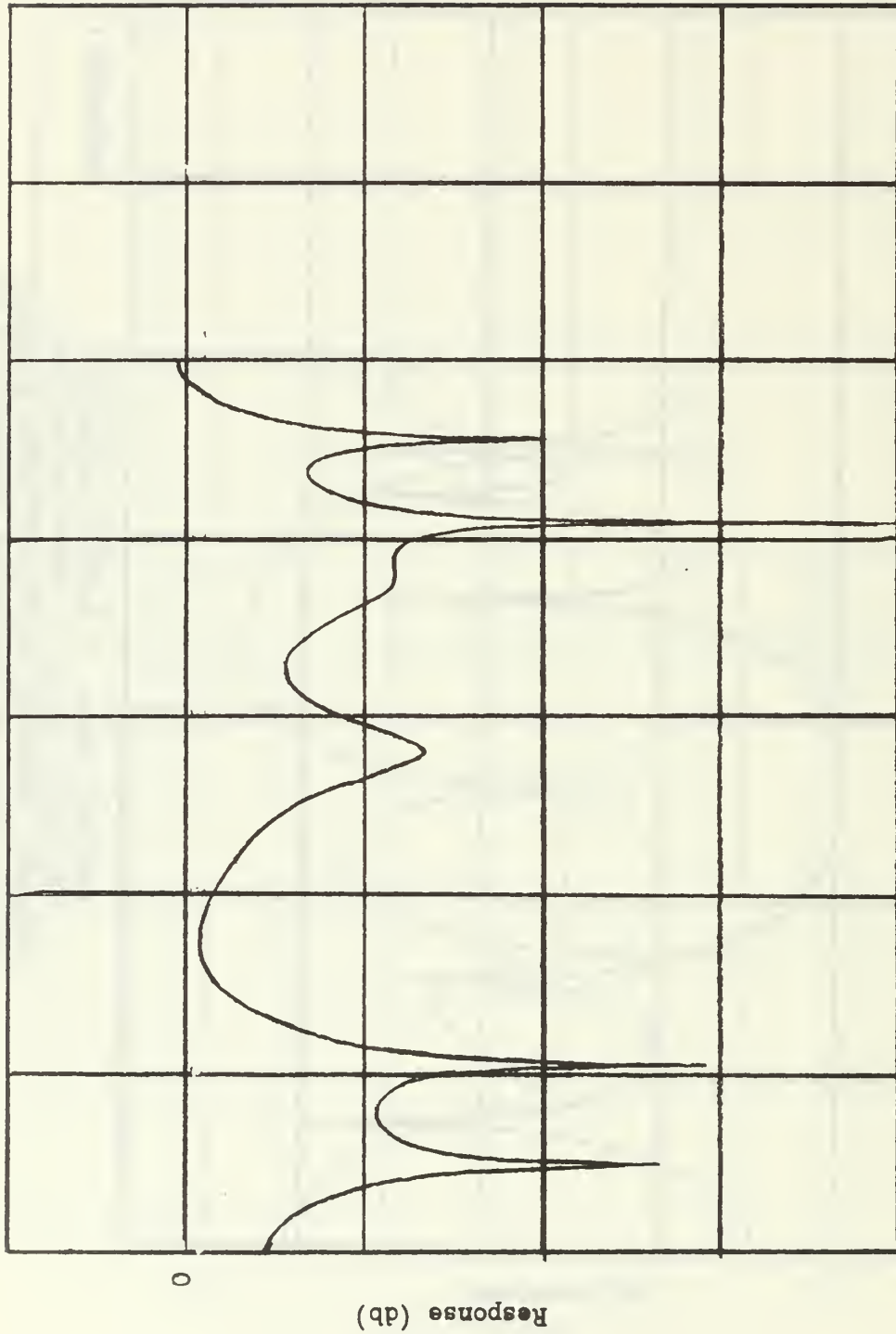


Frequency f

Vertical scale = 2.00E+01 units/inch

Horizontal scale = 2.00E+02 units/inch

Frequency responses for $\theta = 45^\circ$ after adapting for 300 cycles of the desired frequency 500 Hz.

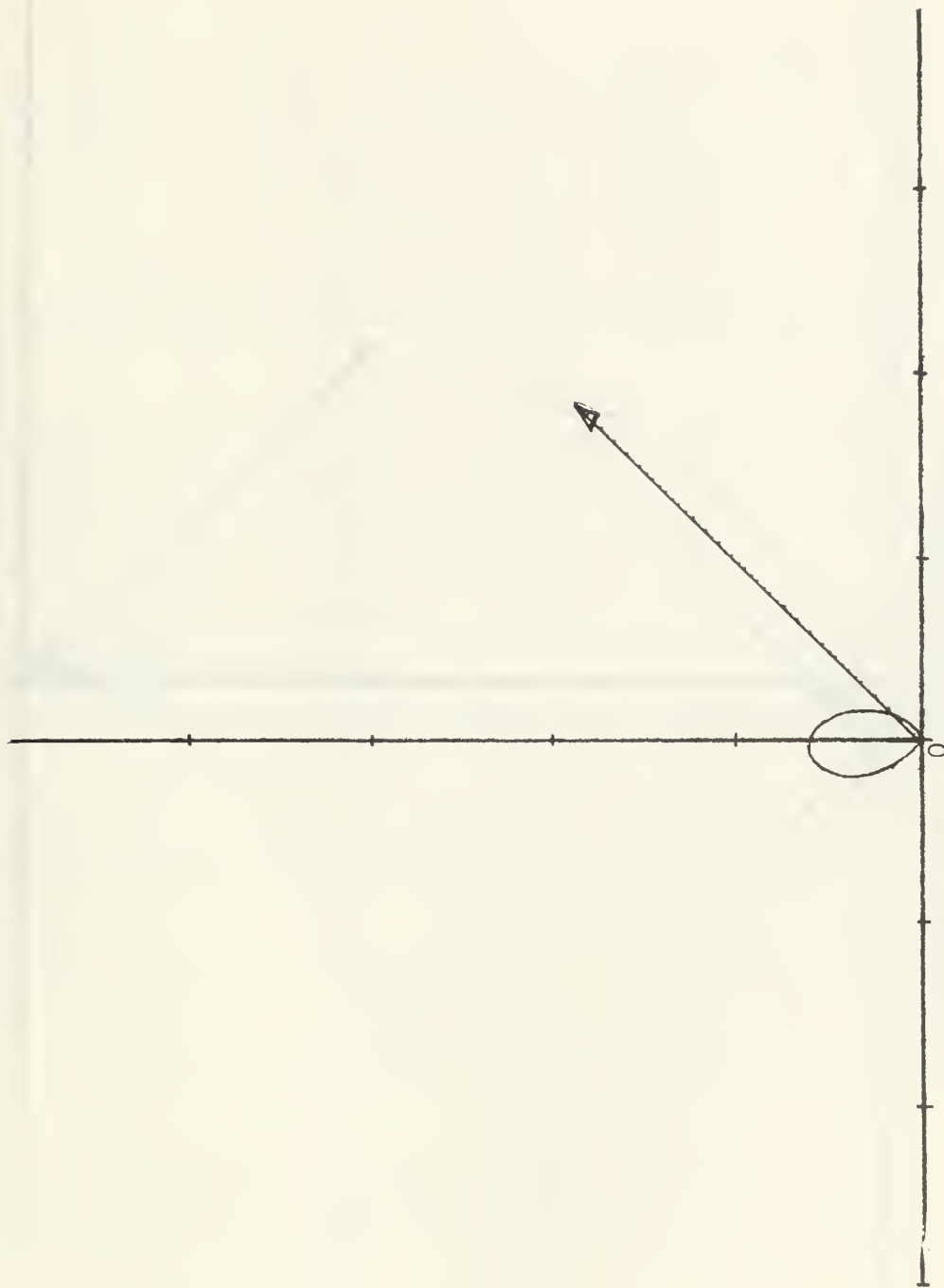


Frequency f

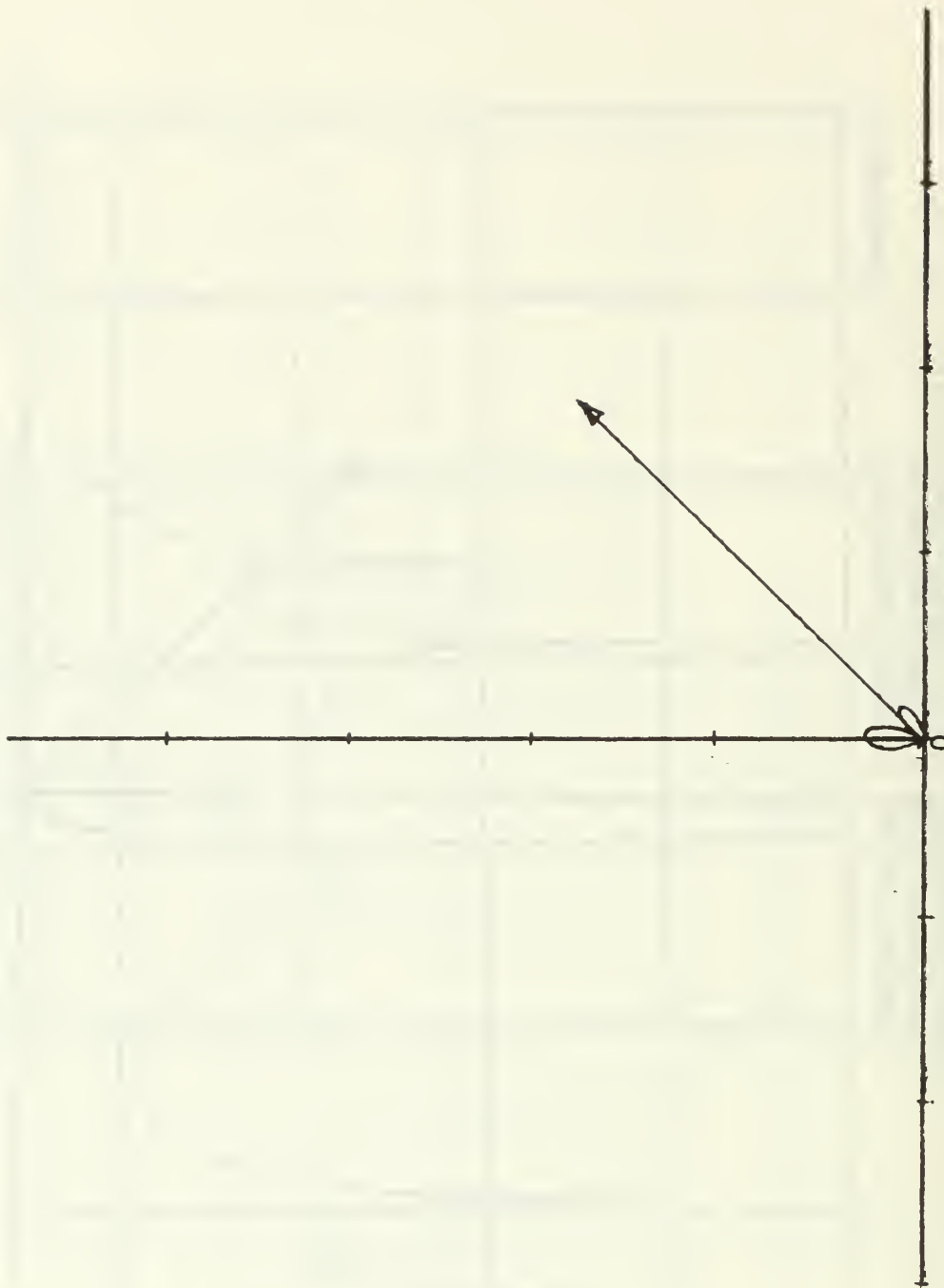
Vertical scale = 2.00E+01 units/inch

Horizontal scale = 2.00E+02 units/inch

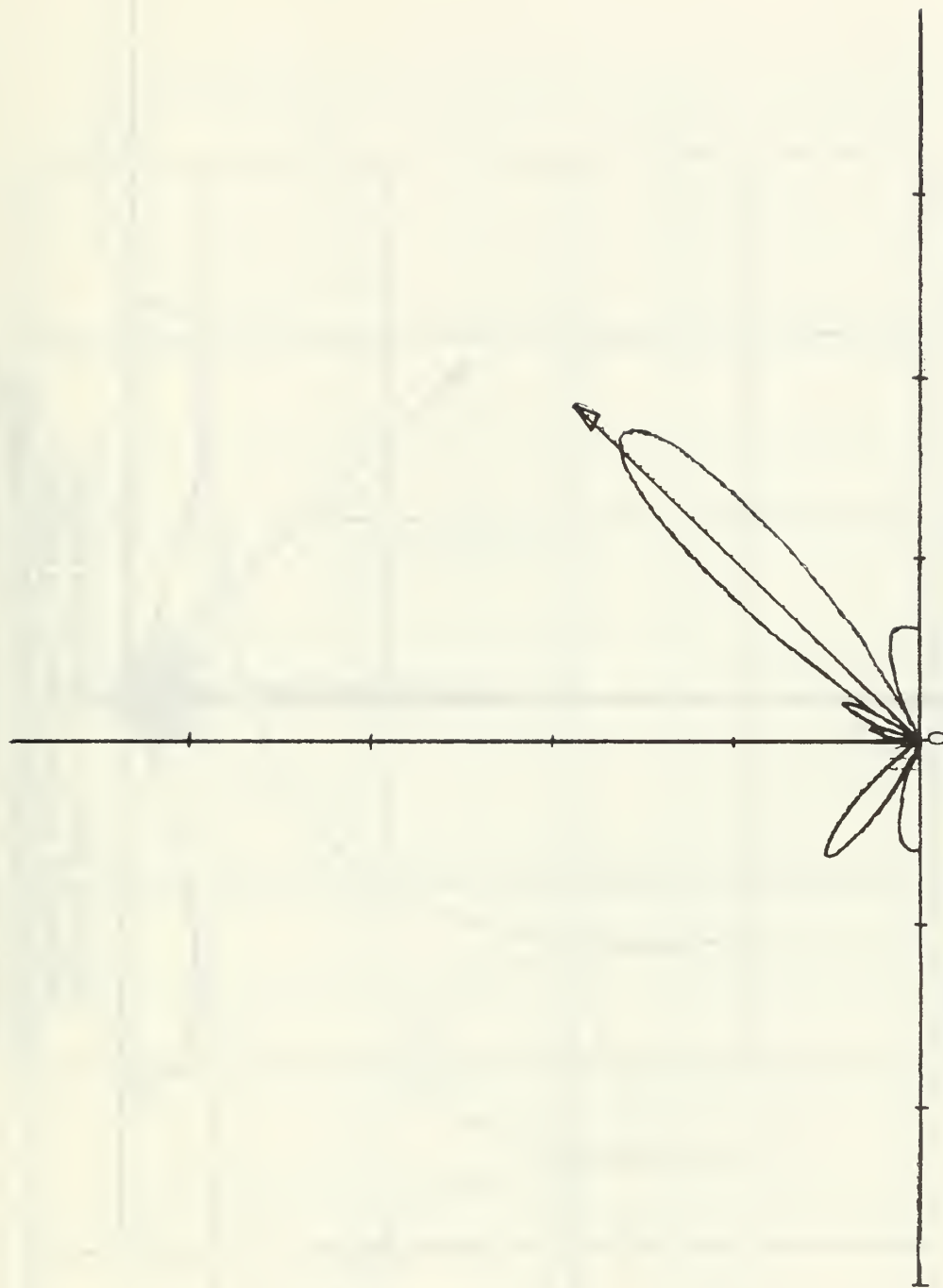
Frequency responses for $\theta = 88.8^\circ$ after adapting for 300 cycles of the desired frequency 500 Hz.



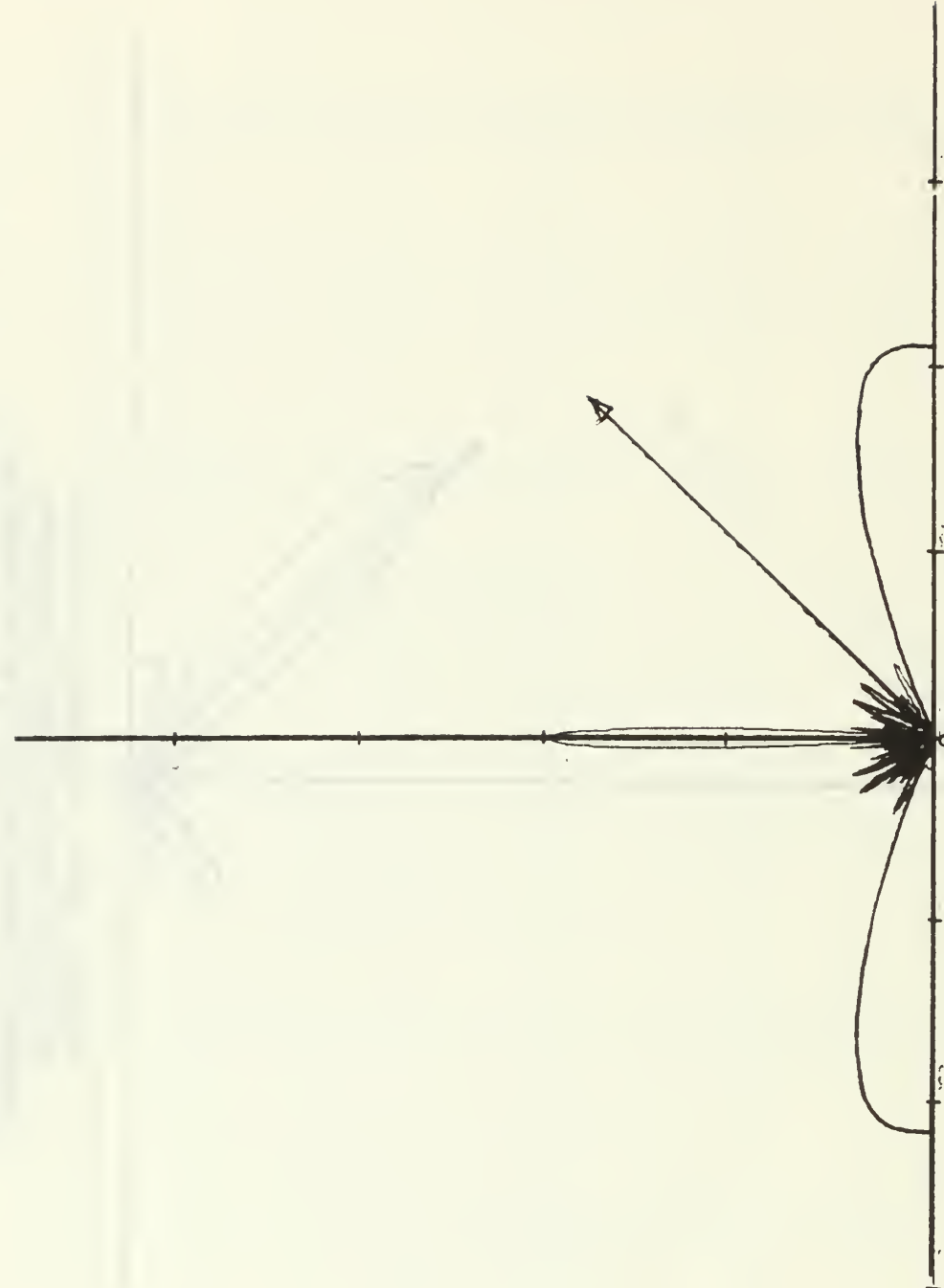
Vertical scale = 4.00E-01 units/inch
Horizontal scale = 4.00E-01 units/inch
Directivity pattern at 100 Hz after adapting
for 600 cycles of the desired frequency 500 Hz.



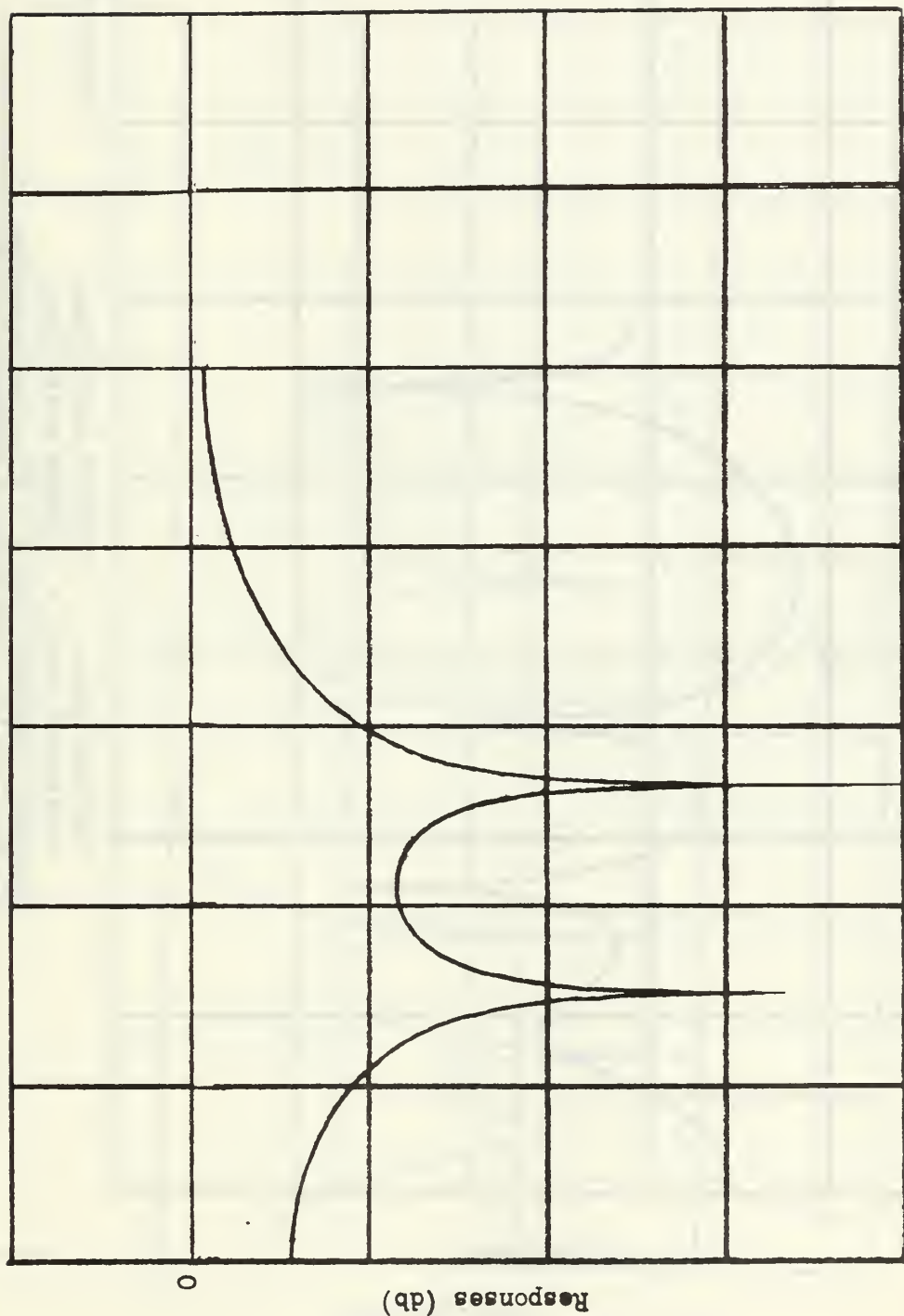
Vertical scale = 4.00E-01 units/inch
Horizontal scale = 4.00E-01 units/inch
Directivity pattern at 200 Hz after adapting
for 600 cycles of the desired frequency 500 Hz.



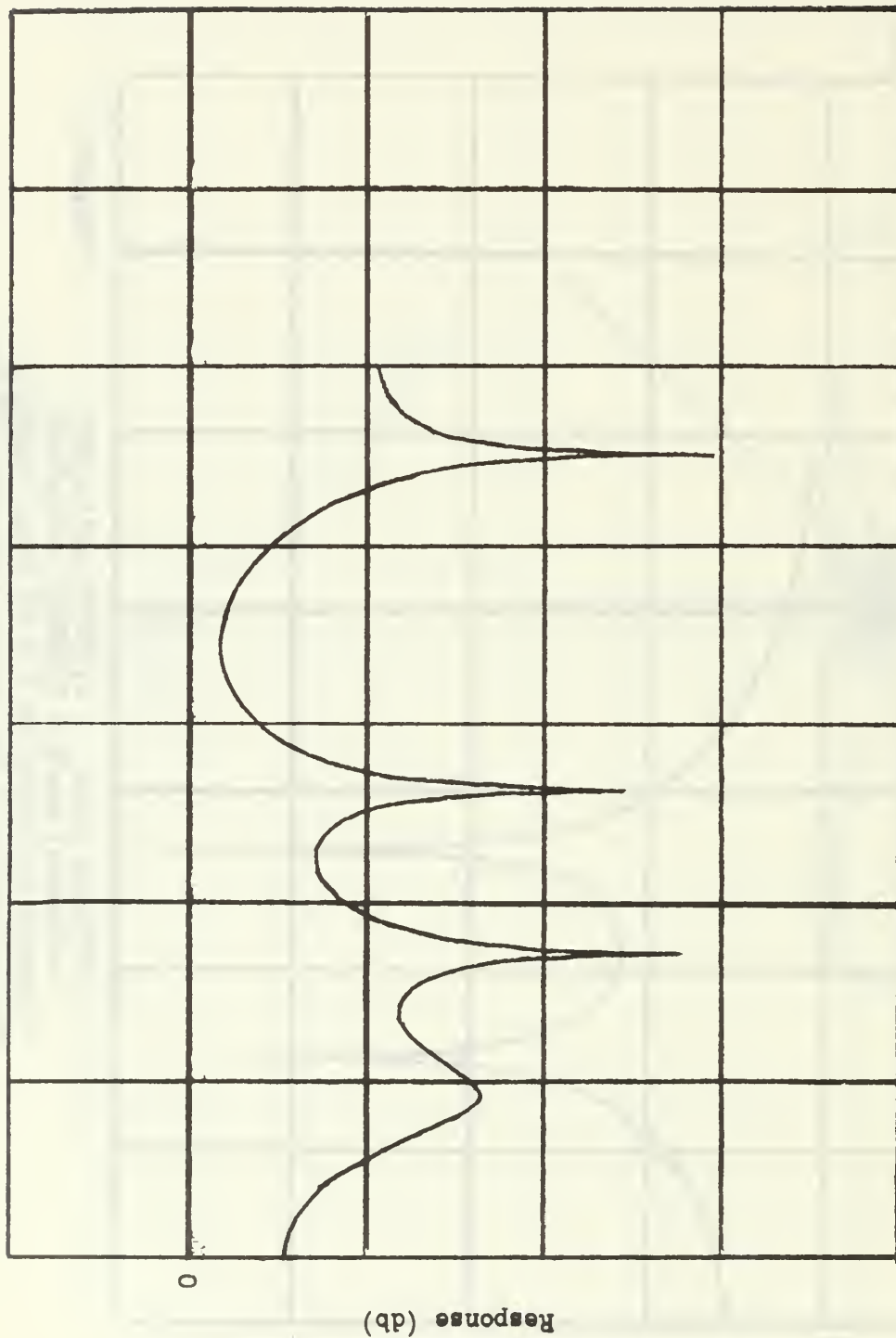
Vertical scale $\approx 4.00\text{E}-01$ units/inch
Horizontal scale $\approx 4.00\text{E}-01$ units/inch
Directivity pattern at 500 Hz after adapting
for 400 cycles of the desired frequency 500 Hz.



Vertical scale = 4.00E-01 units/inch
Horizontal scale = 4.00E-01 units/inch
Directivity pattern at 1000Hz after adapting
for 600 cycles of the desired frequency 500 Hz.



Vertical scale = $2.00E+01$ units/inch
 Horizontal scale = $2.00E+02$ units/inch
 Frequency responses for $\theta = 0^\circ$ after adapting
 for 600 cycles of the desired frequency 500 Hz.

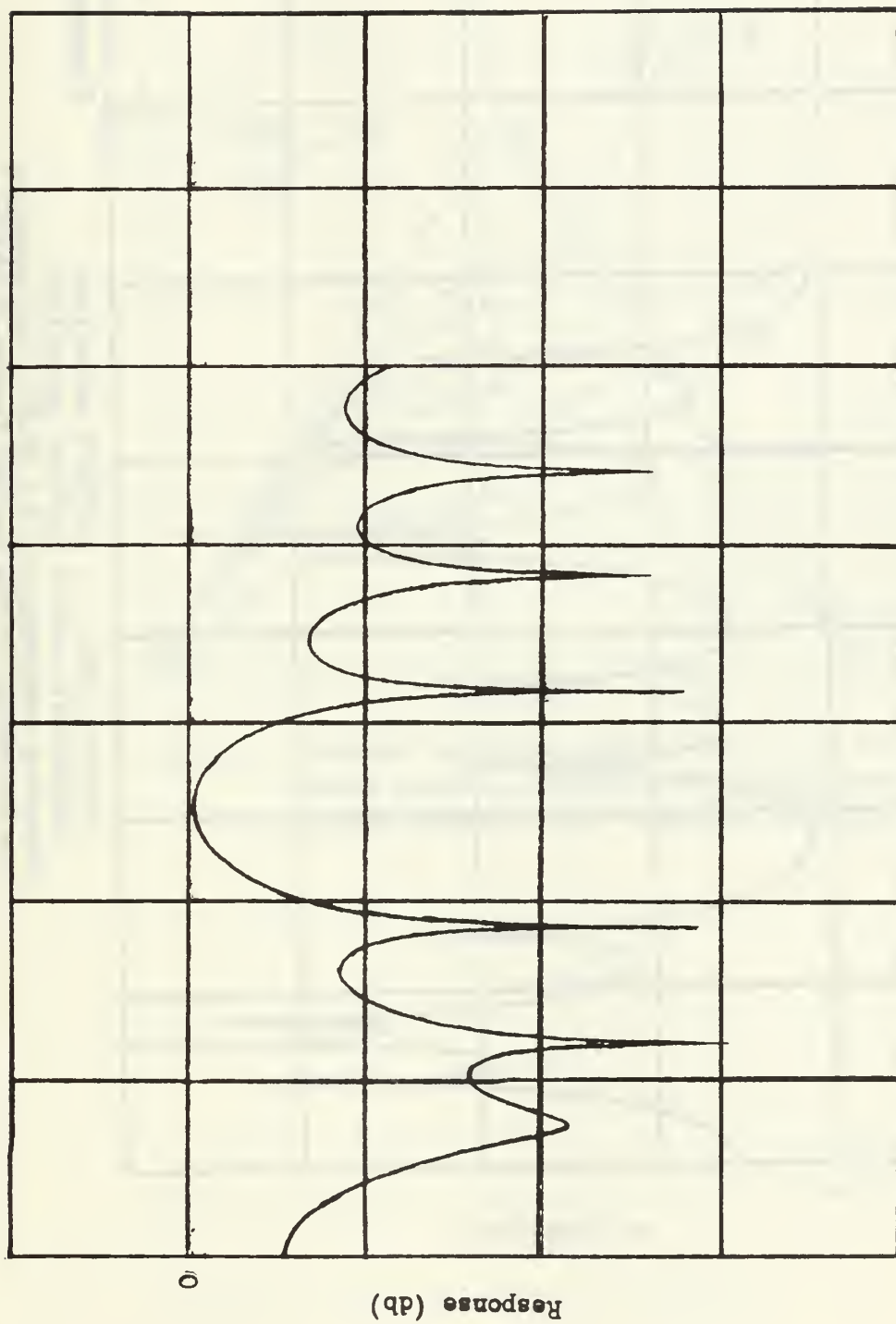


Frequency f

Vertical scale = $2.00E+01$ units/inch

Horizontal scale = $2.00E+02$ units/inch

Frequency responses for $\theta = 30^\circ$ after adapting for 600 cycles of the desired frequency 500 Hz.

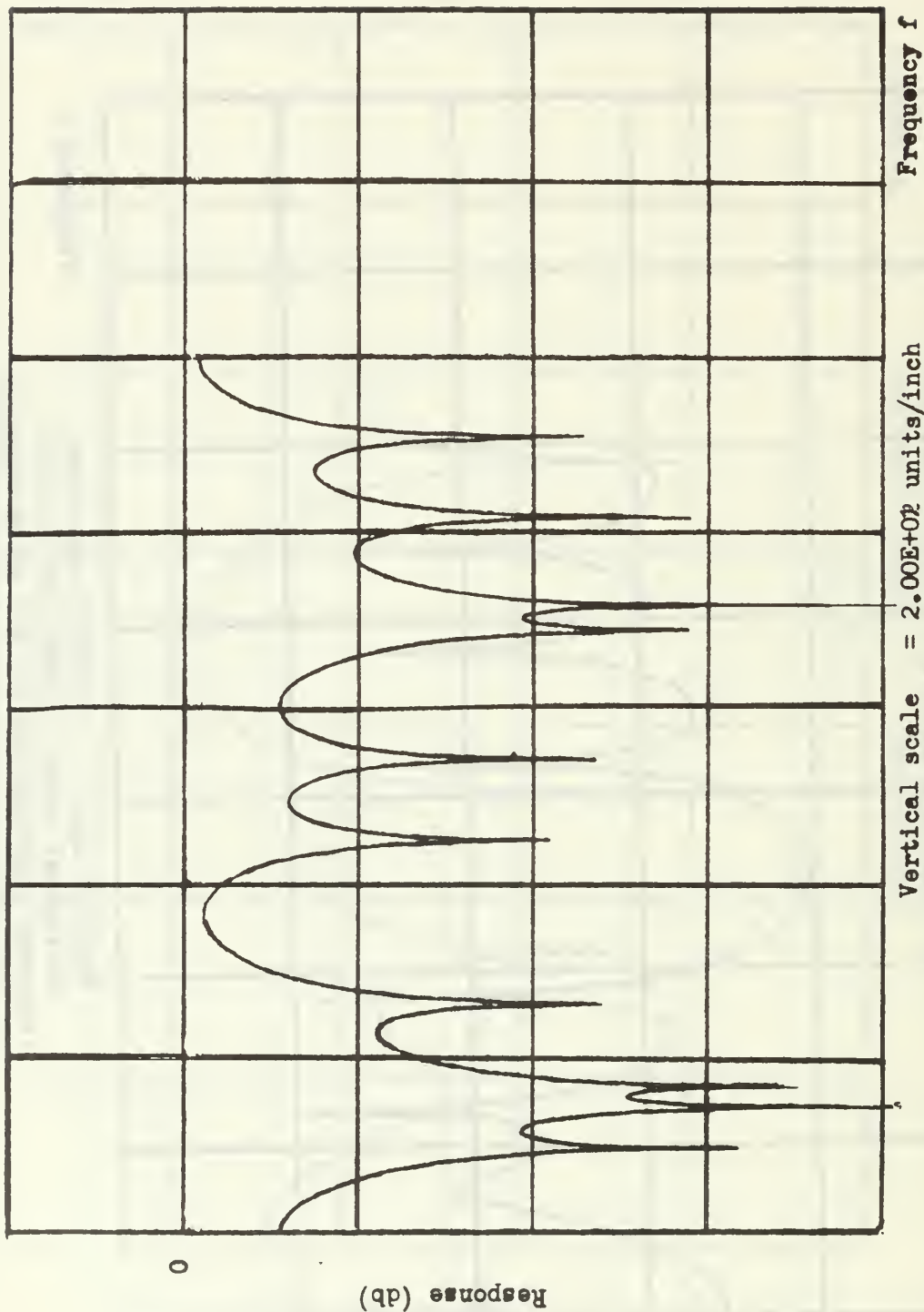


Frequency f

Vertical scale = 2.00E+01 units/inch

Horizontal scale = 2.00E+02 units/inch

Frequency responses for $\theta = 45^\circ$ after adapting for 600 cycles of the desired frequency 500 Hz.



Frequency f

Vertical scale = 2.00E+02 units/inch

Horizontal scale = 2.00E+02 units/inch

Frequency responses for $\theta = 88.8^\circ$ after adapting for 600 cycles of the desired frequency 500 Hz.

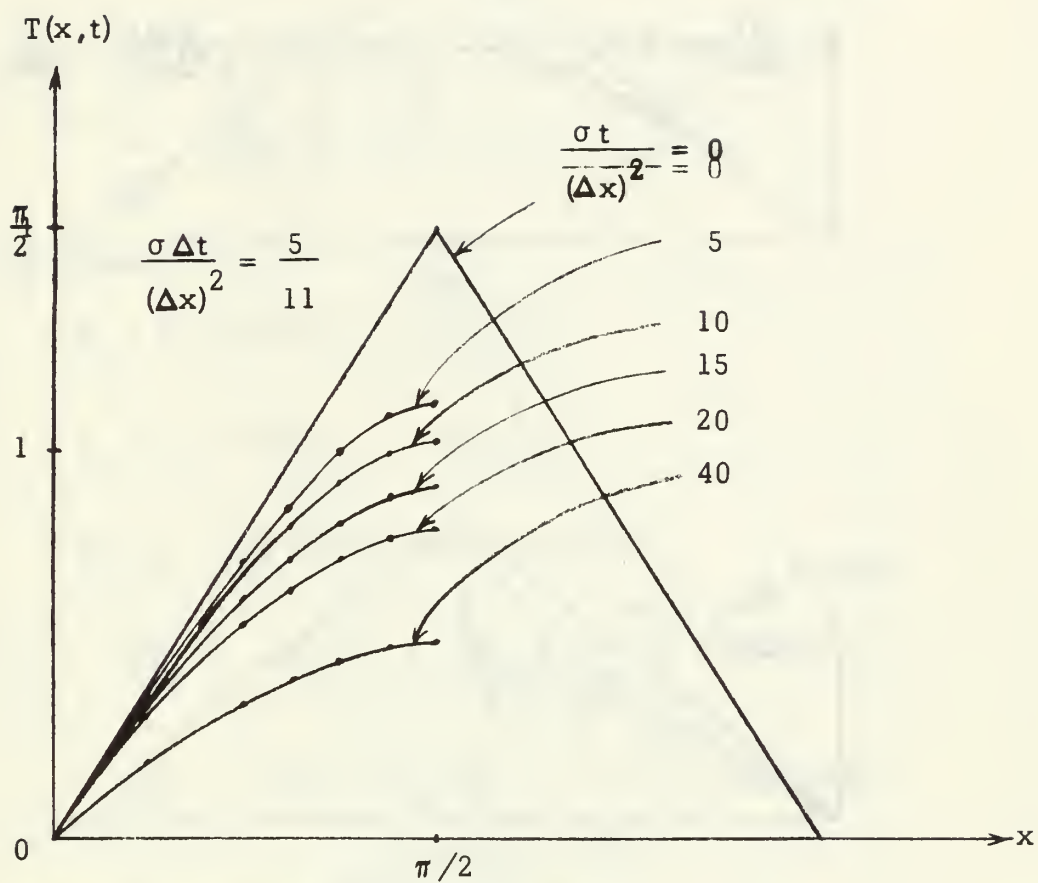


Fig. 1. Diffusion equation solution by Fourier series and stable discrete difference methods. Curves = Fourier solution; dots = discrete solution.

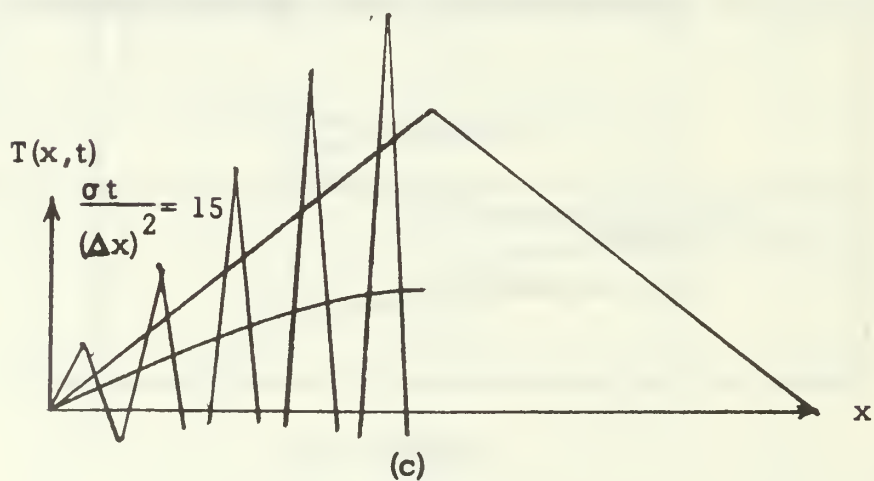
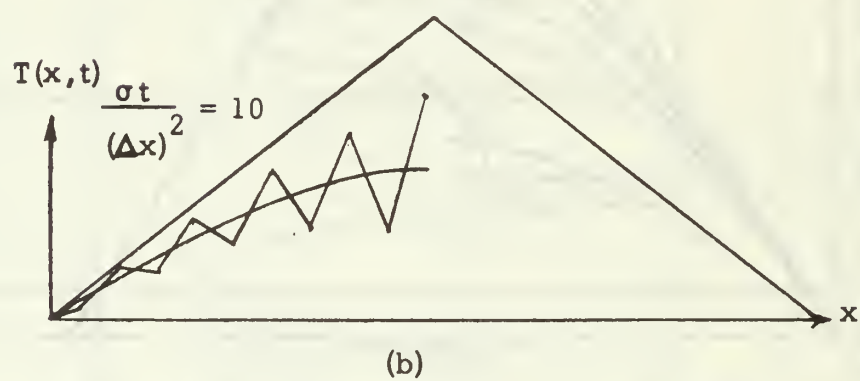
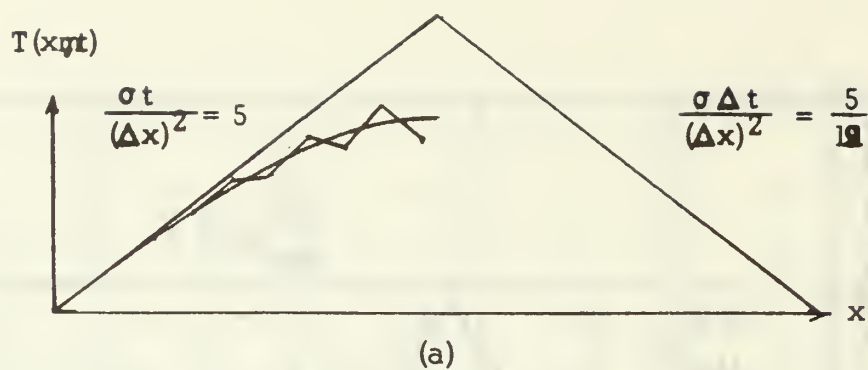
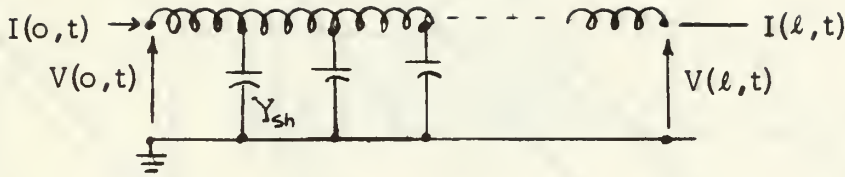


Fig. 2. Instabilities in discrete diffusion equation solution.



For general line

$$Z_s = R_s + j2\pi fL_s + \frac{1}{j2\pi fC_s}$$

$$Y_{sh} = G + j2\pi fC_{sh} + \frac{1}{j2\pi fL_{sh}}$$

$V(x,t)$, $I(x,t)$ satisfy instantaneously

$$\frac{\partial V}{\partial x} = - (R_s I + L_s \frac{\partial I}{\partial t} + \frac{1}{C_s} \int^t I(x, \xi) d\xi)$$

$$\frac{\partial I}{\partial x} = - (G V + C_{sh} \frac{\partial V}{\partial t} + \frac{1}{L_{sh}} \int^t V(x, \xi) d\xi)$$

or for steady state $j2\pi ft$ variations

$$\frac{d\hat{V}(x)}{dx} = - Z_s \hat{I}(x)$$

$$\frac{d\hat{I}(x)}{dx} = - Y_{sh} \hat{V}(x)$$

where

$$V(x,t) = \text{Re}(\hat{V}(x)e^{j2\pi ft})$$

$$I(x,t) = \text{Re}(\hat{I}(x)e^{j2\pi ft})$$

Fig. 3. The One-Dimensional Transmission Line Equations

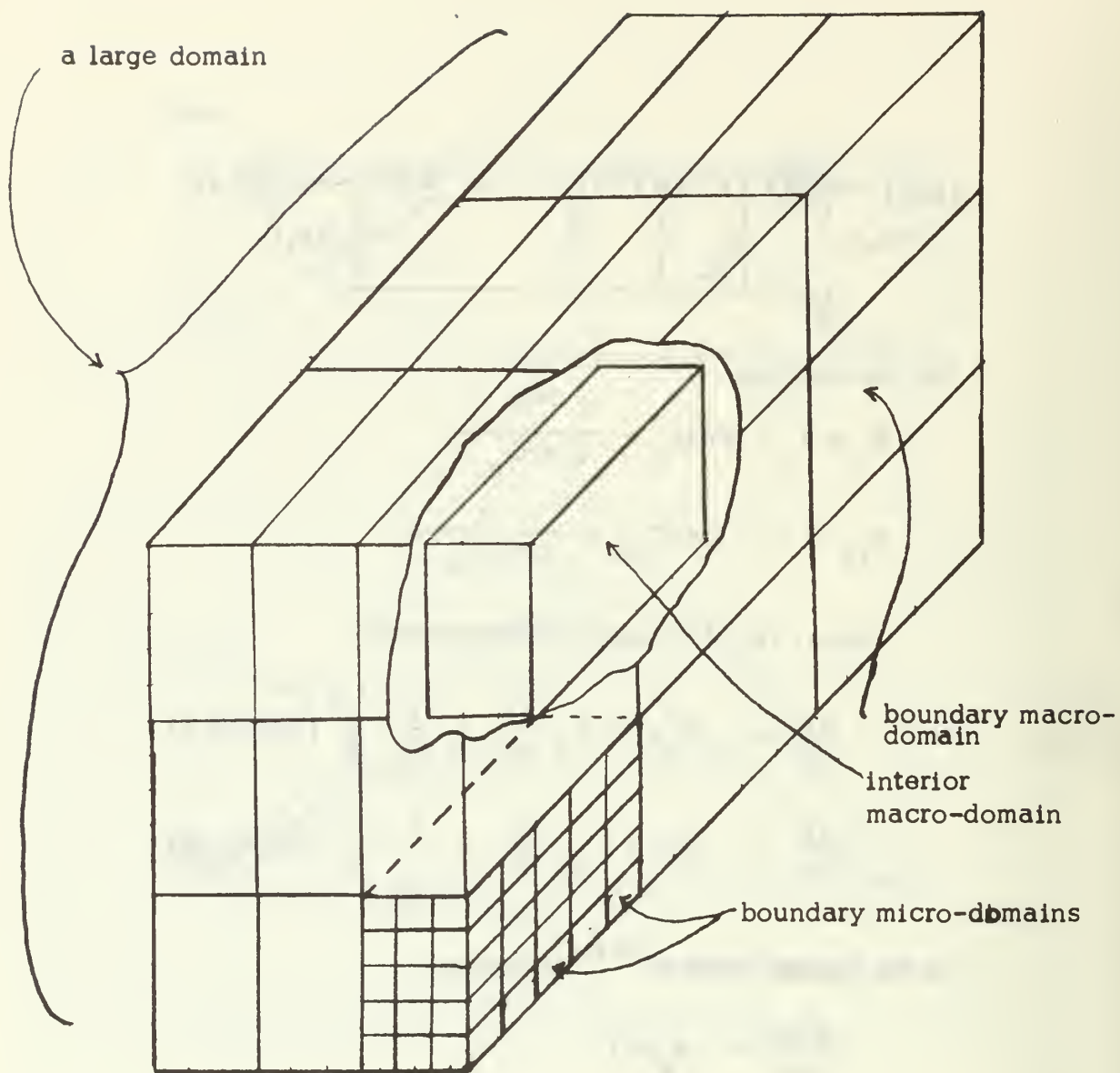


Fig. 4. Geometric model of the integration domains

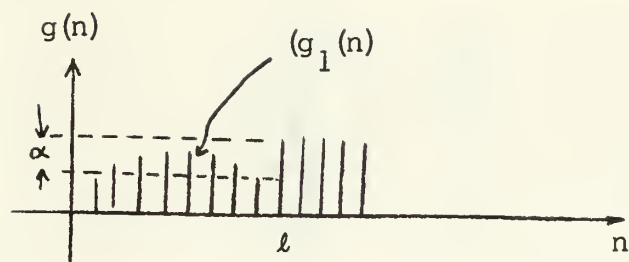
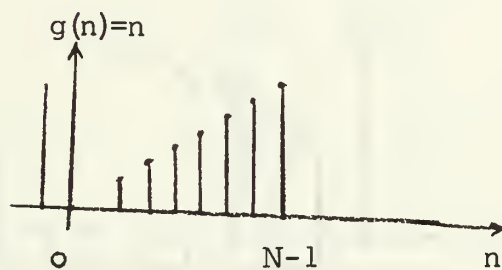
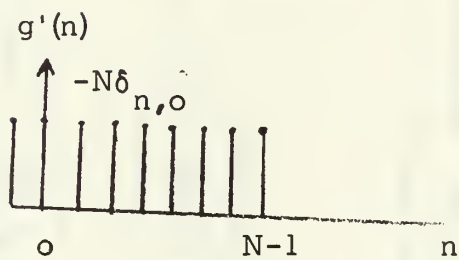


Fig. 5. Function having discontinuity at $n = \ell$.



(a)

function $g(n) = n$



(b)

Fig. 6. Derivative of function $g(n) = n$

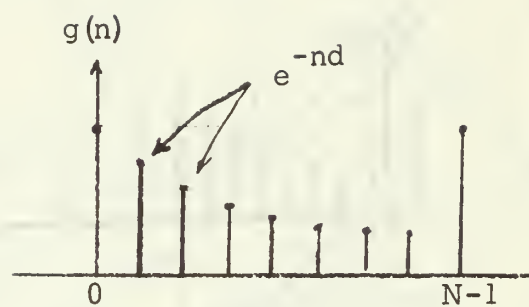


Fig. 7. Recursive example: The exponential function

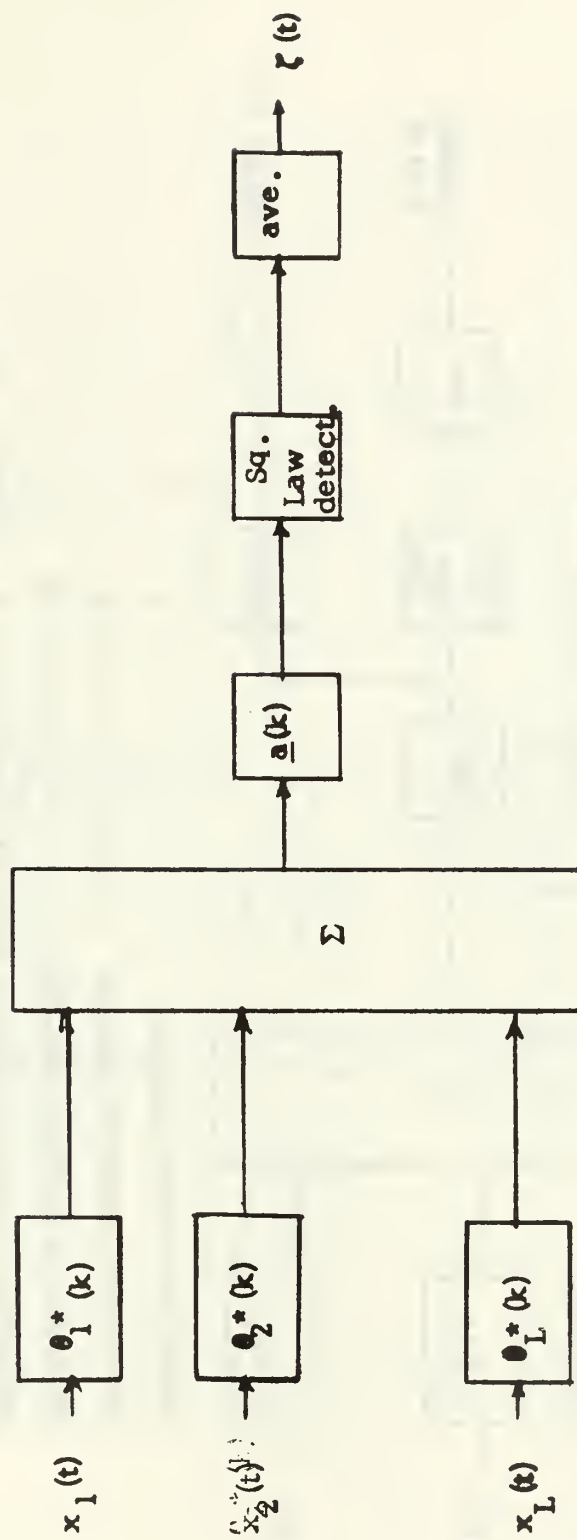


Fig. 9. The quadratic-cost-function optimal-estimator Array Processor.

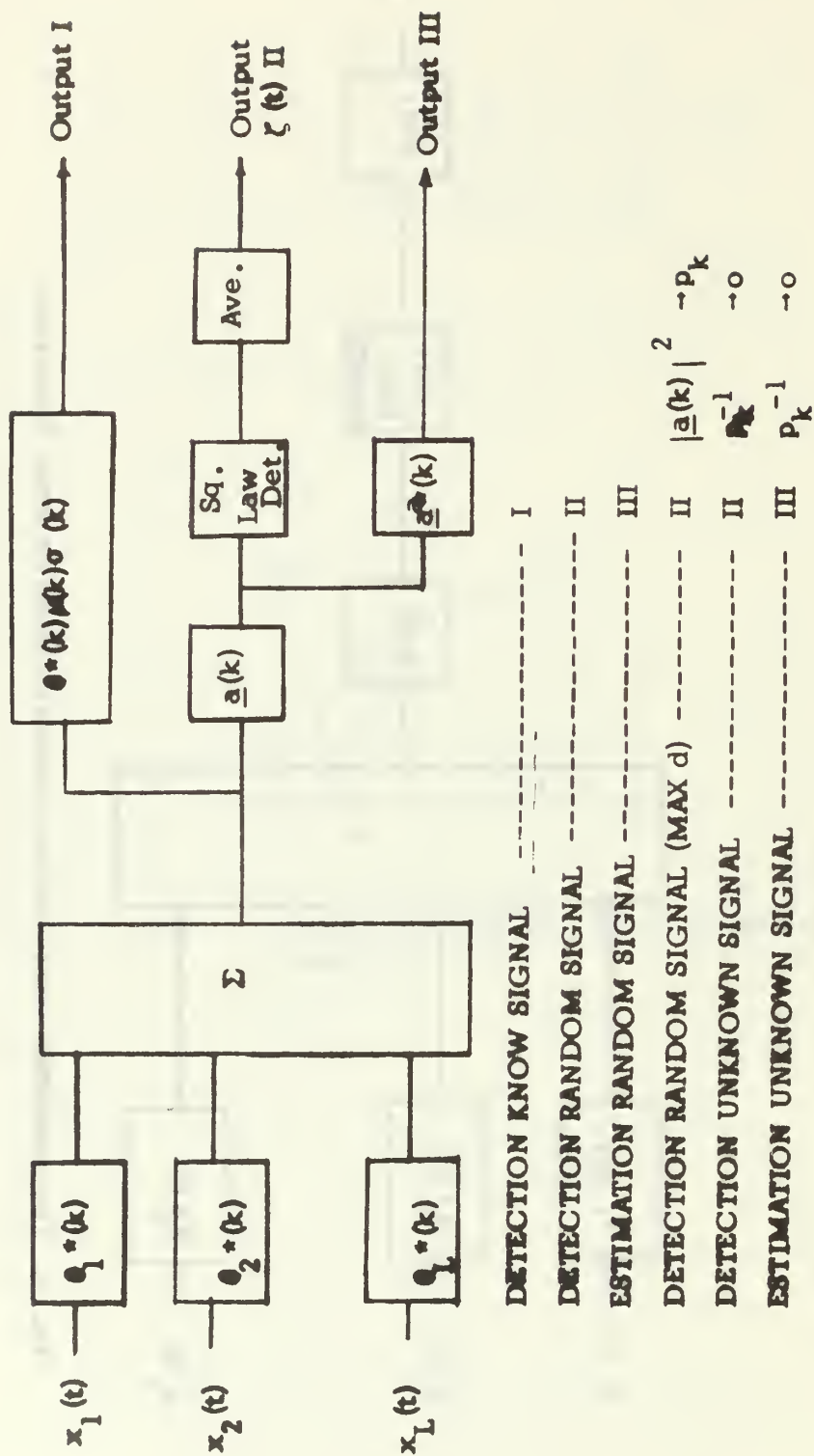


Fig. 10. The general quadratic-cost-function optimal-estimator array processor.

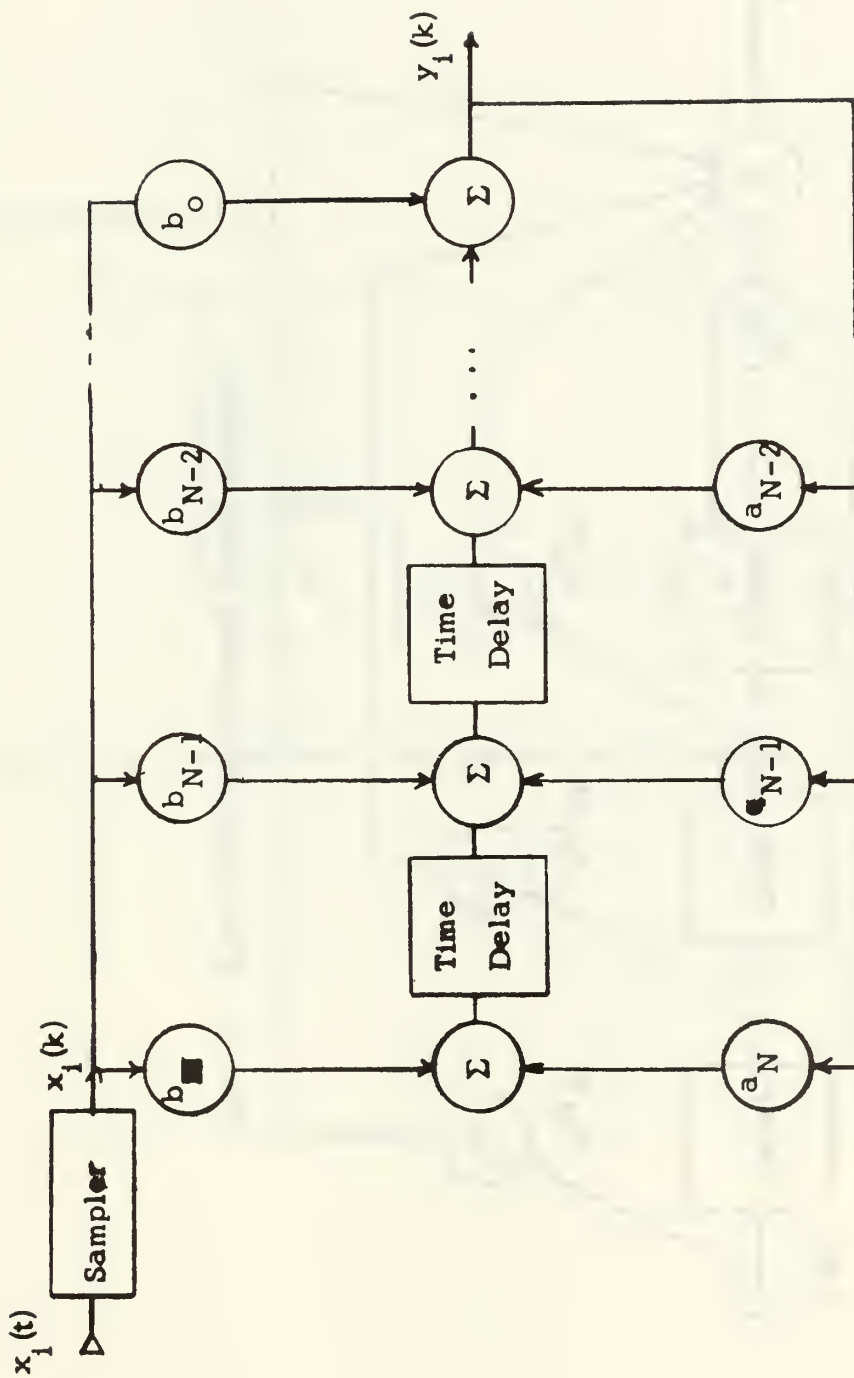


Fig. 11. General feedback, feed-forward sampled-data network

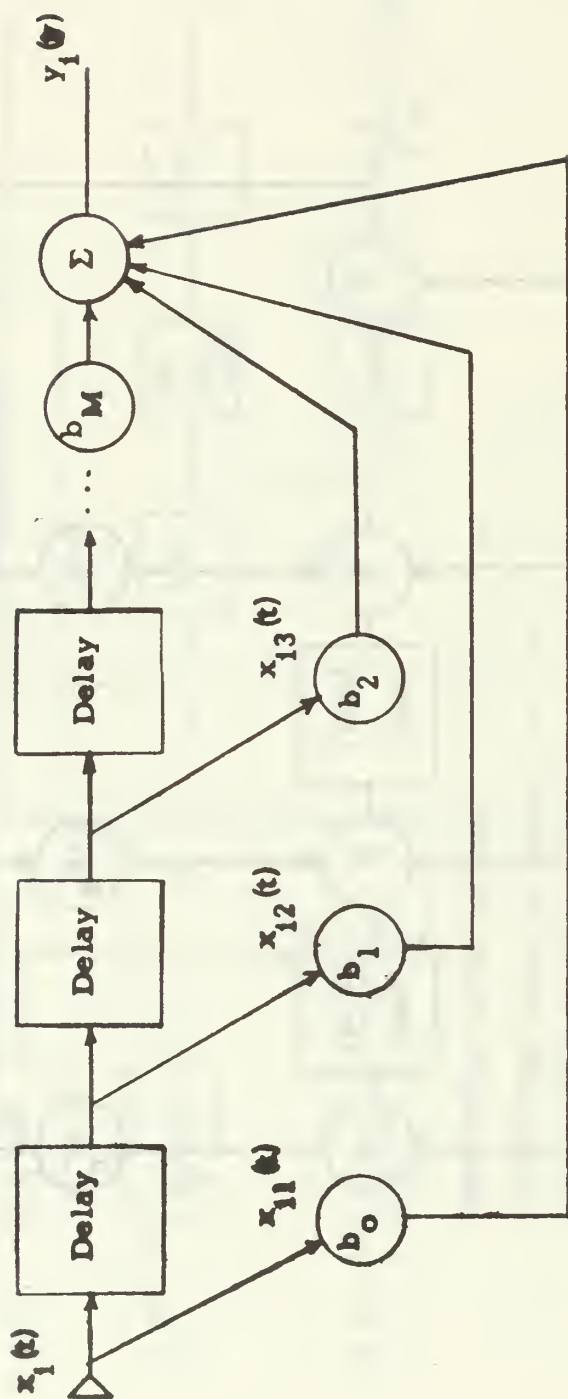


Fig. 12. Wide-band feed-forward delay network.

$$x(t) = x(o) \cos(2\pi f_1 t + \phi_o)$$

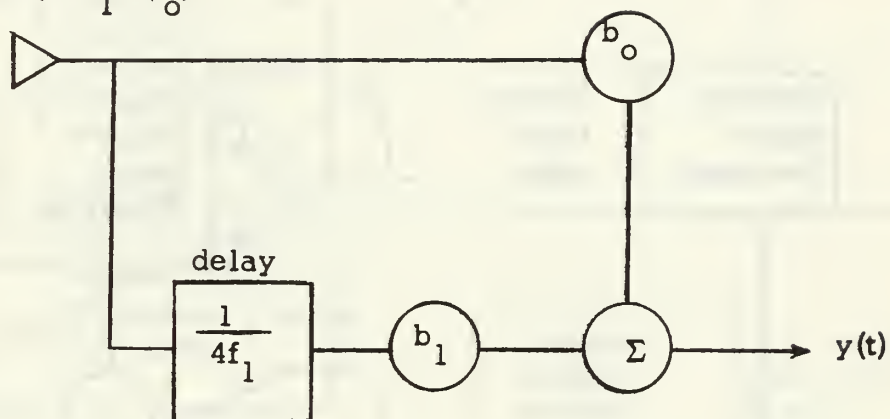


Fig. 13. Narrow-band feed-forward delay network

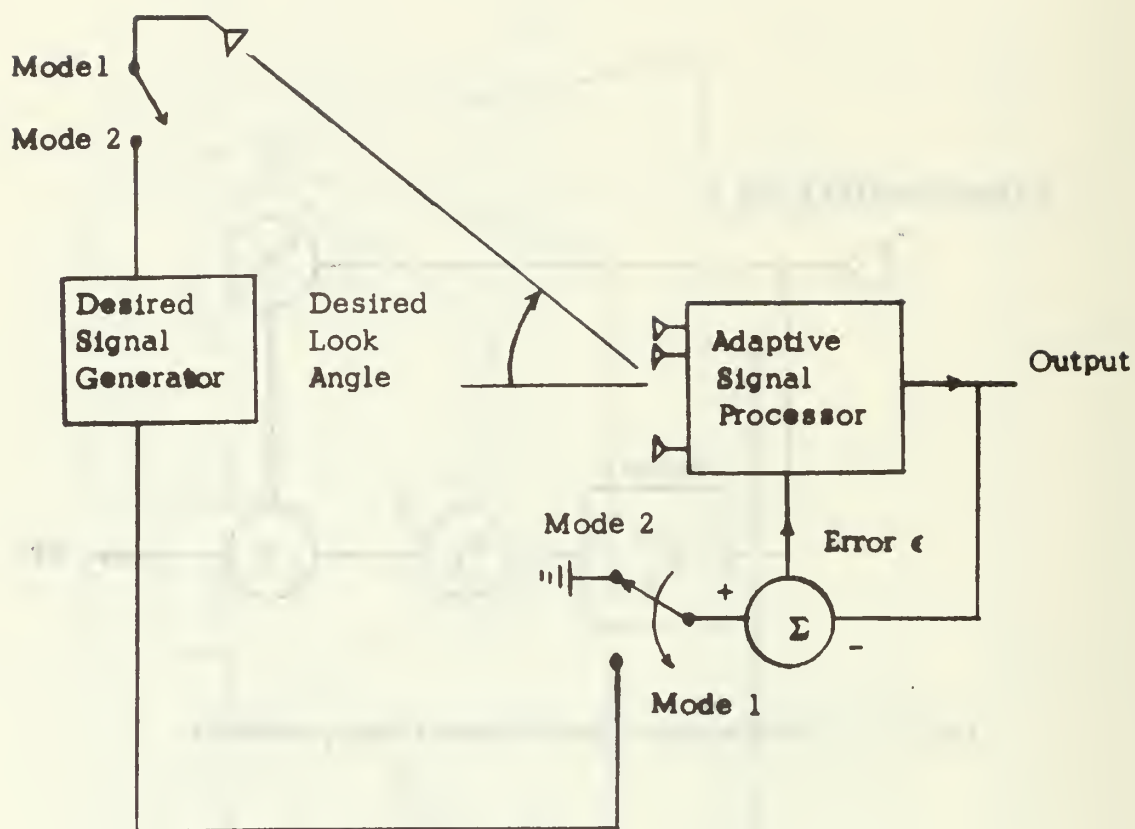


Fig. 14. The on-off adaptive network scheme

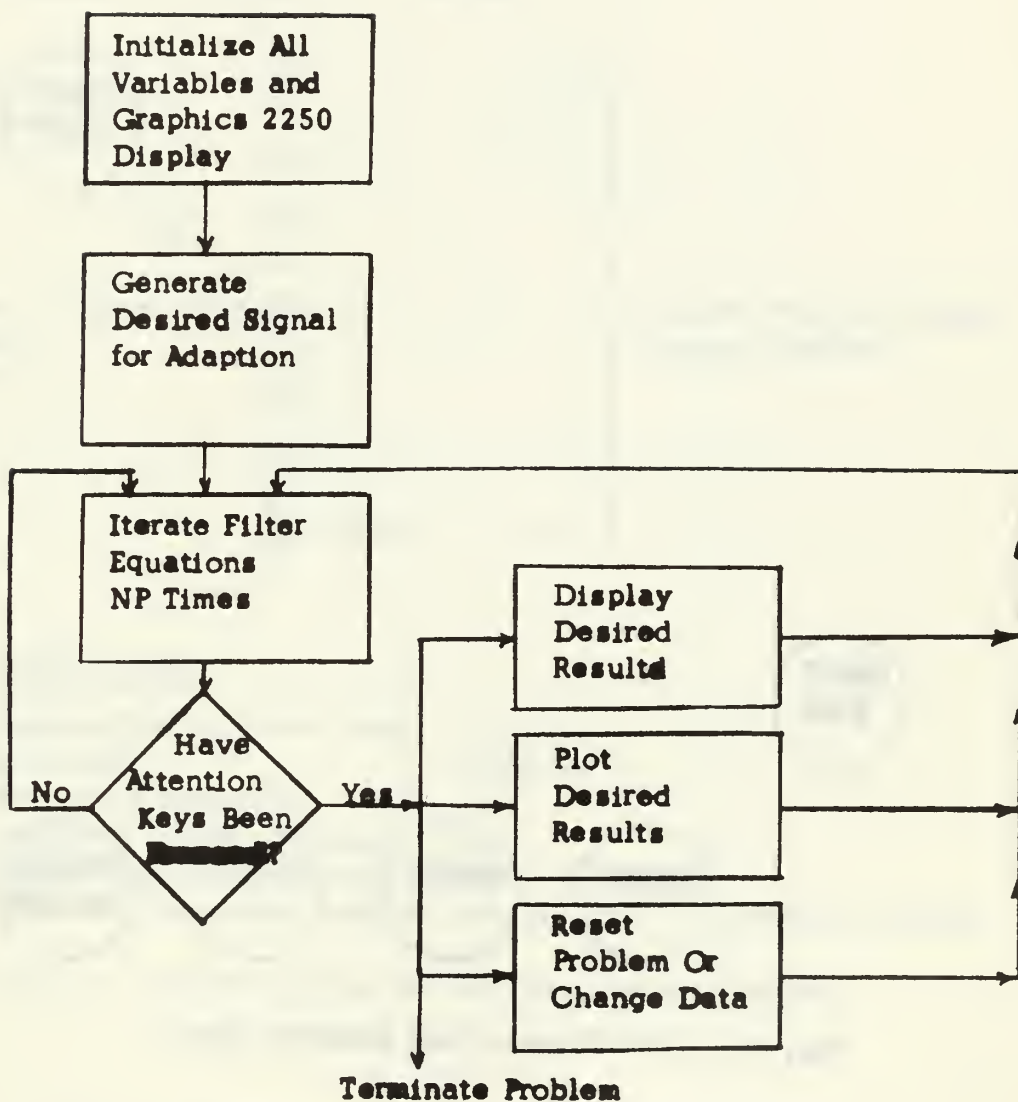


Fig. 15. Flow Diagram of Array Simulation Program .

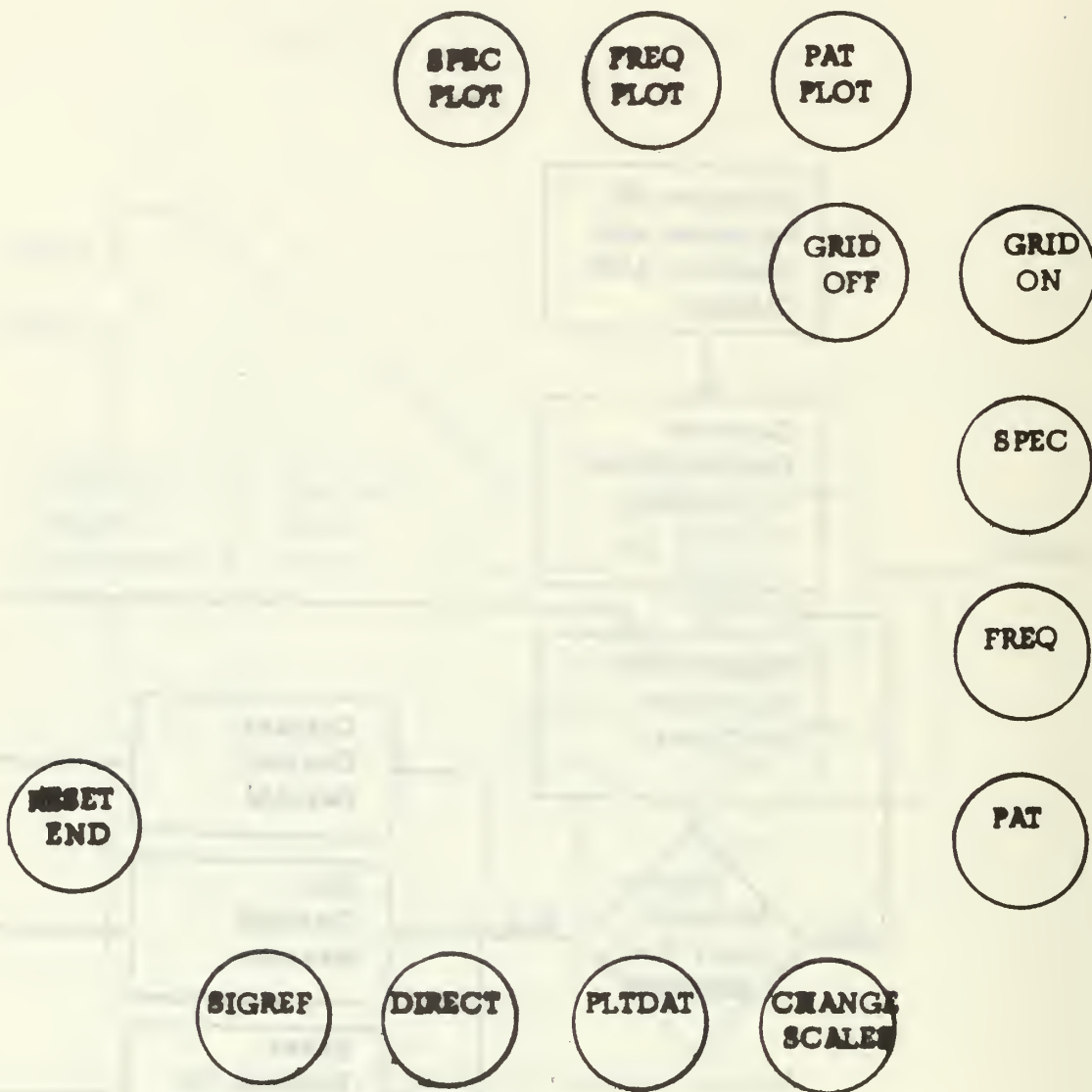
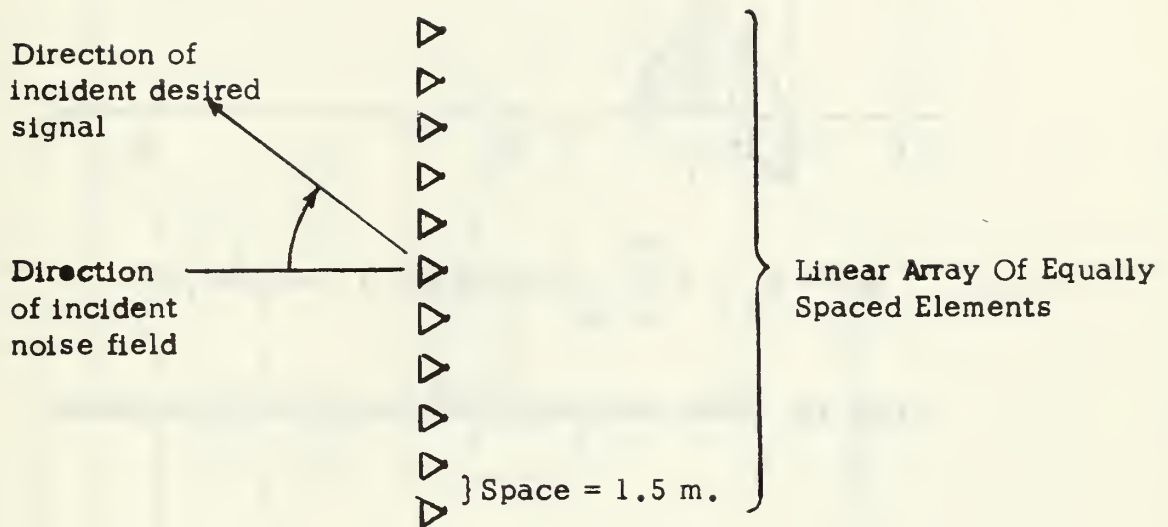


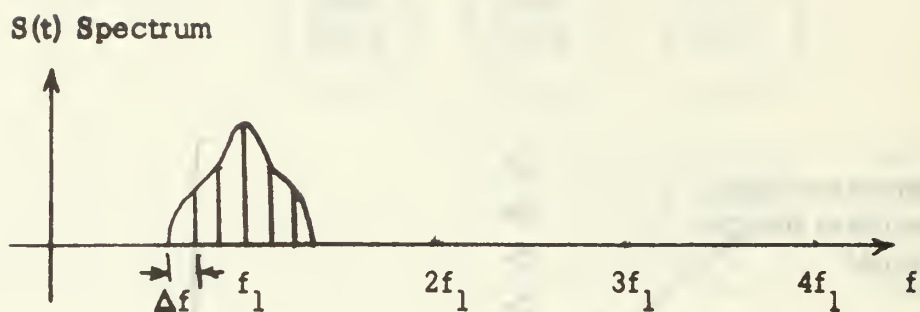
Fig. 16. 2250 Display Unit Attention Keys.



Problem Data:

Desired Signal Center Freq. = 500 Hz
 Bandwidth = 200 Hz
 Number of Elements = 11
 Number of Delays/Element = 2
 Pattern Plots at Angles of 0° , 30° , 45° , 90°
 Frequency Response Plots at 100 Hz, 250 Hz, 500 Hz, 1000 Hz

Fig. 17. Physical Layout of Array Simulation Program.



$$S(t) = A_o \left[1 + \sum_{k=-M}^{+M} \cos(2\pi(k \Delta f) t) \right] \cos(2\pi f_1 t)$$

Fig. 18. Wide-band Amplitude Modulated Input Signal.

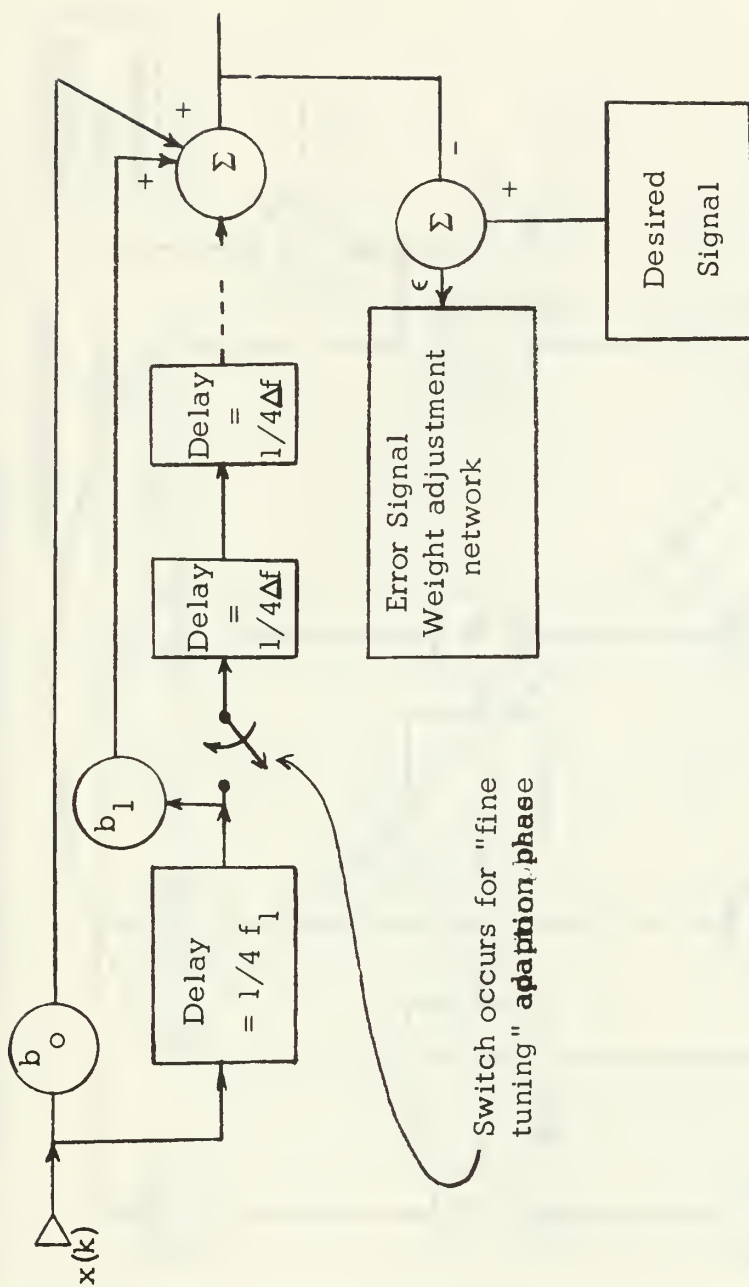


Fig. 19. Adaption network with variable delays.

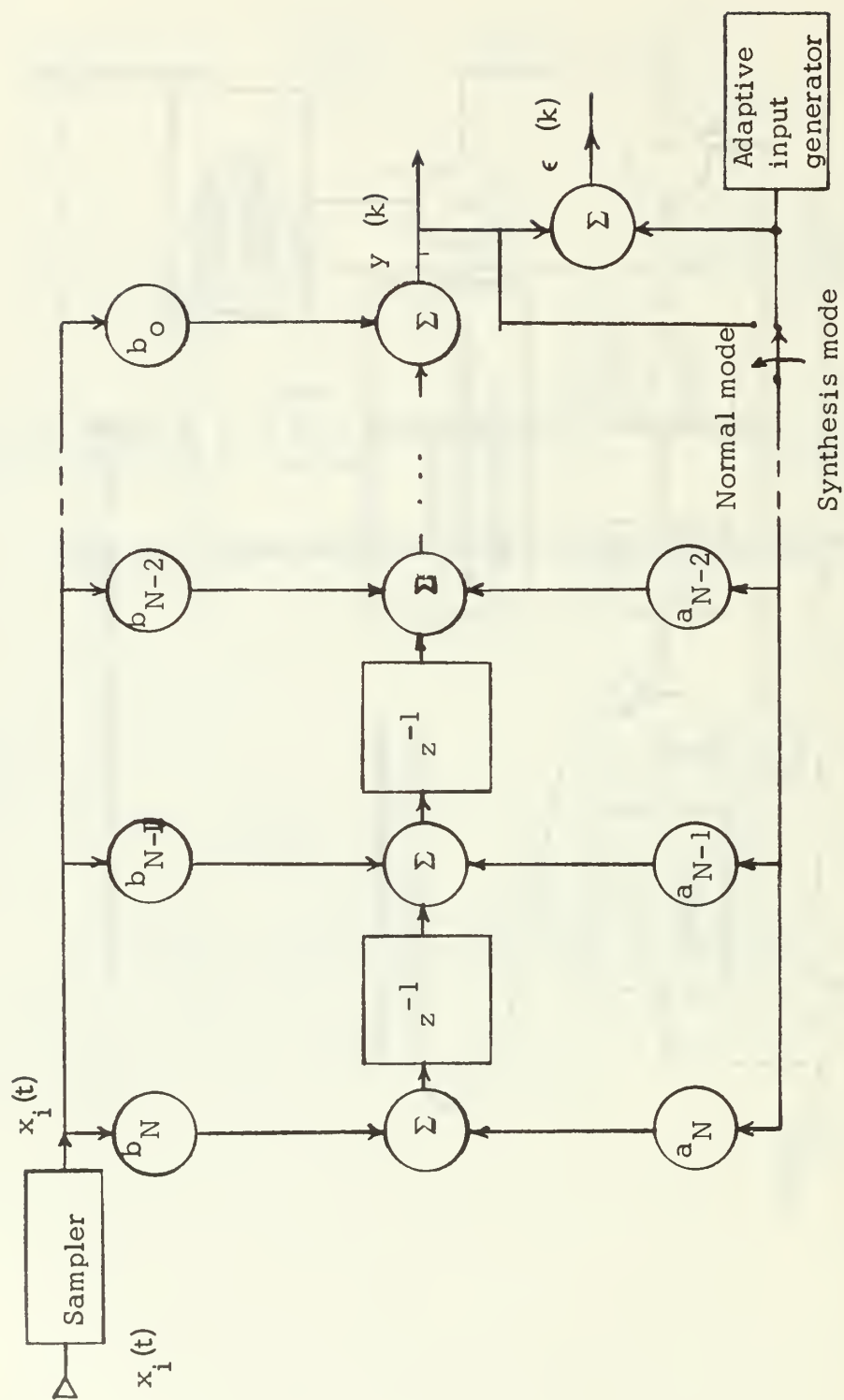


Fig. 20. Mantey's modified adaptive feedback network.

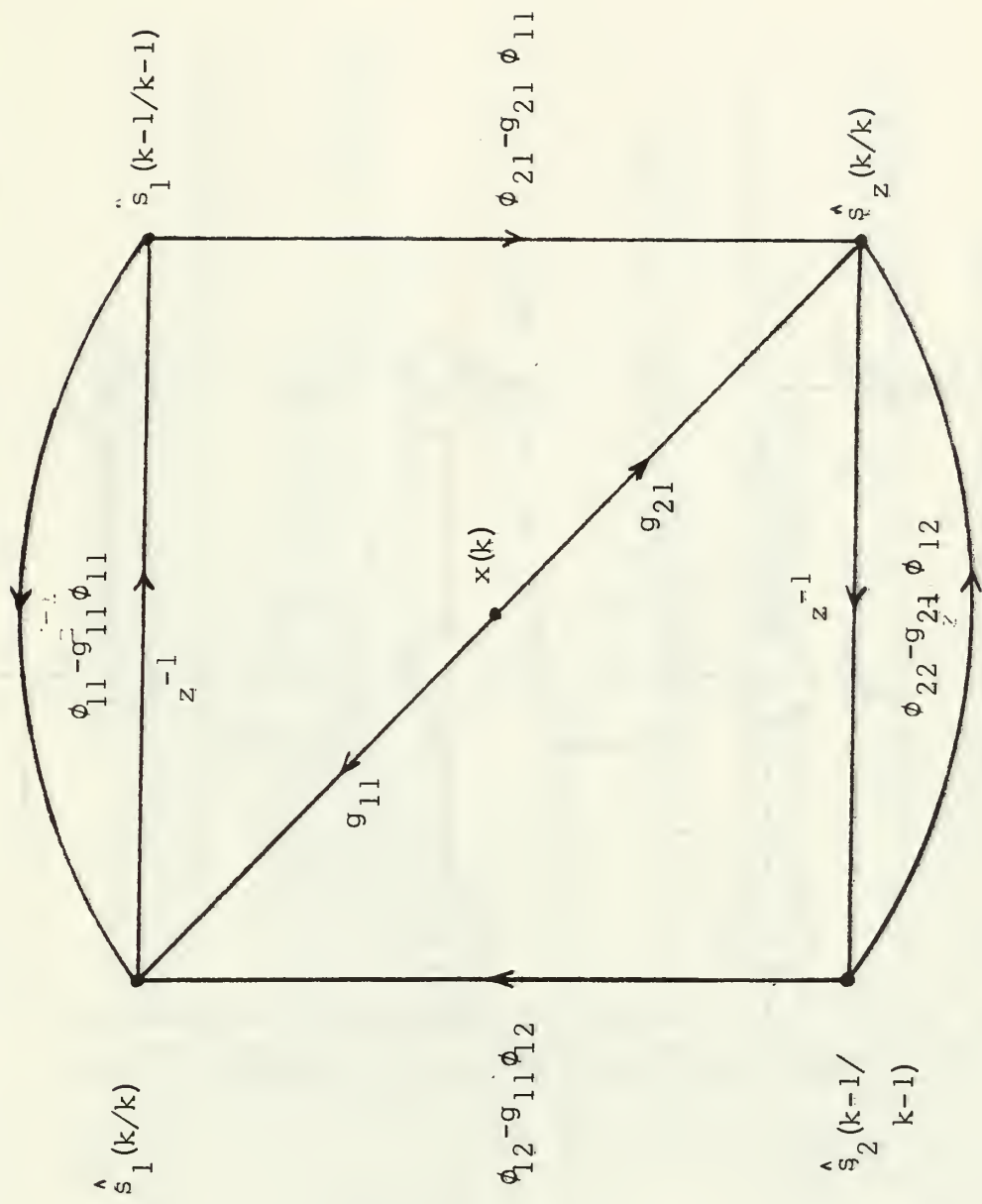


Fig. 20. Two-delay Kalman Filter State Transition Diagram.

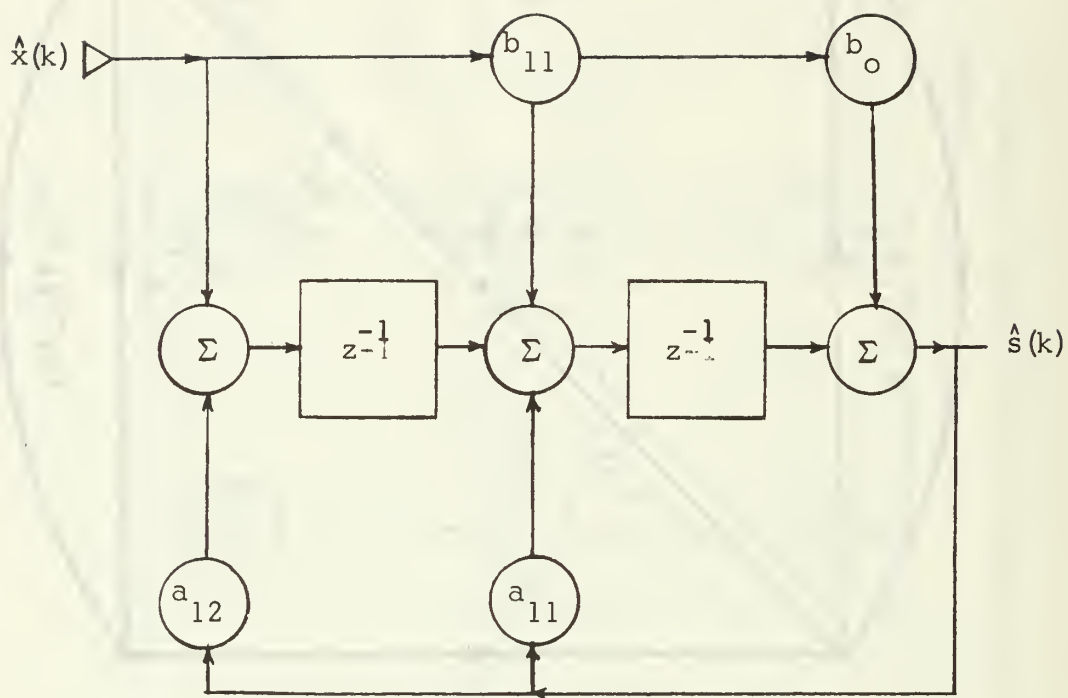


Fig. 22. Two-delay Kalman Filter Adaptive network.


```

BLOCK DATA
REAL T/O.0/,DT,DT1/O.0005/,DT2/O.0005/,DT3/O.0005/
COMMON/TDT/T,DT,DT1,DT2,DT3
INTEGER NX/20/,NY/25/,NT,DT1,DT2,DT3
COMMON/SIZE/NX,NY,NT,IIS1,IIS2
REAL R/G.0/,LUO/6.66667E-04/,LU(25),G/O.0/,CO/1.333333E-04/,C(25),
&DX/5.0/,DY/5.0/
COMMON/PARAM/R,LUO,LU,G,CO,C,DX,DY
REAL EX/8.0/,EY/8.0/,SLOPE/O.707/,FACT/1.0/
COMMON/SCALE/EX,EY,SLOPE,FACT
INTEGER IU/O/,IR/O/,IXL/2/,IYL/2/,IW/6/,IH/4/,IG/O/
COMMON/DRW/IU,IR,IXL,IYL,IW,IH,IG
INTEGER IWR/100/,IDRW/100/,IEN/5/,IANL/100/,ISAMP/16/,IDFT/100
&/,NWV/1/,JDET/20/,ITYPE/1/
COMMON/PGMC/IWRT,IDRW,IEN,IANL,ISAMP,IDFT,NWV,JDET,ITYPE
INTEGER II/1/,NARB/O/,NARE/25/
COMMON/STF/II,NARB,NARE
REAL DV(25,25,3)/1875*0.0/
COMMON/DERIV/DV
END
COMPLEX*8DA(10,16)/160*(0.0,0.0)/
REAL*8ITI(10,12)
REAL LUC
REAL LU(25),C(25)
REAL LX(25),LY(25),LXI(25),LYI(25),CXY(25),CXYI(25),CXY2(25),
&CXY2I(25),CXY4(25),CXY4I(25),RLU(25),GC(25)
REAL*4LA(100)
REAL V(25,25,3)/1875*0.0/
REAL*4ITSPEC(24)/96HARRAY SPECTRAL COEF. Y(I,J)=ALOG10(DA(I,J))
&REAL*8JOHNSCN BOX 70
&REAL*8IARG(12)/96HFOURIER PHASE OF ITH RECEIVER
&REAL*8JOHNSCN BOX 70
REAL LARG(10)/40H0 1 2 34 5 6 78 9
REAL LAF(10)/40H0 1 2 34 5 6 78 9
REAL X(16)/0.0,1.0,2.0,3.0,4.0,5.0,6.0,7.0,8.0,9.0,10.0,11.0,12.0
&13.0,14.0,15.0/
COMMON/TDT/T,DT,DT1,DT2,DT3
COMMON/SIZE/NX,NY,NT,IIS1,IIS2
COMMON/PGMC/IWRT,IDRW,IEN,IANL,ISAMP,IDFT,NWV,JDET,ITYPE
COMMON/PARAM/R,LUO,LU,G,CO,C,DX,DY
COMMON/STF/II,NARB,NARE
COMMON/BARG/R(16),ARG(16),NI
COMMON/TITLE/ITI,LA

```

← END OF BLOCK DATA

```

COMMON/SCALE/EX,EY,SLOPE,FACT
COMMON/DRW/IU,IR,IXL,IYL,IW,IH,IG
COMMON/EL/LX,LXI,LV,LVI,CXY,CXYI,CXY2,CXY4,CXY4I,RLU,GC
DIMENSION IREAD(10),IWRITE(10)
DIMENSION EA(40),EXF(40),FA(40),FX(40)
DIMENSION Y(16)

C
C NX=NO. OF INCREMENTS INTO WHICH THE X COORDINATE IS DIVIDED
C
C NY=NO. OF INCREMENTS INTO WHICH THE Y COORDINATE IS DIVIDED
C
TIME=ITIME(0)*0.01
CALL CANCEL(2)
READ(5,10)IREAD,IWRITE
WRITE(6,20)NX,NY,NT,II S1,II S2,IREAD,IWRITE
WRITE(6,260)T,DT1,DT2,DT3
WRITE(6,170)R,G,DX,DY
WRITE(6,180)IU,IR,IXL,IYL,IW,IH,IG
WRITE(6,200)IWRIT,IIDRW,IEN,IANL,ISAMP,IDFT,NWV
WRITE(6,210)EX,EY,SLOPE,FACT
WRITE(6,240)II,NARB,NARE
WRITE(6,30)DT1,NT
READ(5,60)((I,I,I,J),J=1,12),I=1,10)
WRITE(6,70)((I,I,I,J),J=1,12),I=1,10)
READ(5,130)((LA(I),I=1,100)
WRITE(6,140)((LA(I),I=1,100)
READ(5,130)((EA(I),I=1,40),(FA(I),I=1,40),(EXF(I),I=1,40),(FX(I),I=1,40)
WRITE(6,220)((EA(I),I=1,40),(FA(I),I=1,40),(EXF(I),I=1,40),(FX(I),I=1,40)
CALL ICOND(V)
WRITE(6,190)
WRITE(6,160)(LX(I),LXI(I),LV(I),LVI(I),CXY(I),CXYI(I),I=1,NX)
WRITE(6,280)
WRITE(6,160)(CXY2(I),CXY2I(I),CXY4(I),CXY4I(I),RLU(I),GC(I),I=1,NX)
}
DT=DT1
CALL GEN(V)
ISAMP2=ISAMP/2
DO 39 II=1,NT
IM1=II-1
IF(IM1.EQ.1)GO TO 1
IF(MOD(IM1,IWRT))3,1,3
1 WRITE(6,40)T
DO 2 I=1,NX
2 WRITE(6,IWRITE)((I,J,K,V(I,J,K),K=1,NWV),J=1,NY)
WTIME=ITIME(0)*0.01-WTIME

```



```

CXY(I)=1.0/(C(I)*DXY)
CXYI(I)=0.5/CXY(I)
CXY2(I)=2.0*CXY(I)
CXY2I(I)=0.5/CXY2(I)
CXY4(I)=4.0*CXY(I)
CXY4I(I)=0.5/CXY4(I)
GC(I)=G/C(I)
RLU(I)=R/LU(I)
2 CONTINUE
RETURN
END

```

```

SUBROUTINE VEQUAS(V)
COMMON/DT/T,DT,DT1,DT2,DT3
COMMON/SIZE/NX,NY,NT,IIS1,IIS2
REAL LX(25),LXI(25),LY(25),LYI(25),CXY(25),CXYI(25),CXY2(25),
&CXY2I(25),CXY4(25),CXY4I(25),RLU(25),GC(25)
COMMON/EL/LX,LXI,LY,LYI,CXY,CXYI,CXY2,CXY2I,CXY4,CXY4I,RLU,GC
COMMON/PGMC/IWRT,IDRW,IEN,IANL,ISAMP,IDFT,NWV,JDET,ITYPE
REAL DV(25,25,3)
COMMON/DERIV/DV
DIMENSION V(25,25,3)
NXM1=NX-1
NYM1=NY-1
1111 DV(1,1,1)=-CXY4(1)*(V(1,1,2)+V(1,1,3))-GC(1)*V(1,1,1)
1112 DV(1,1,2)=LX(1)*(V(1,1,1))-V(1,2,1))-RLU(1)*V(1,1,2)
1113 DV(1,1,3)=LY(1)*(V(1,1,1))-V(2,1,1))-RLU(1)*V(1,1,3)
1911 DV(1,NY,1)=(LX(1)/CXY4(1))*DV(1,NYM1,2)
1912 DV(NX,1,1)=CXY4(NX)*(V(NXM1,1,3))-V(NX,1,2))-GC(NX)*V(NX,1,1)
1913 DV(NX,1,2)=LX(NX)*(V(NX,1,1))-V(NX,2,1))-RLU(NX)*V(NX,1,2)
DV(NX,1,3)=0.
DV(NX,NY,1)=(LX(NX)/CXY4(NX))*DV(NX,NYM1,2)
IF(ITYPE)200,200,100
DO 1000 I=2,NXM1
1001 DV(I,1,1)=CXY2(I)*(V(I-1,1,3)-V(I,1,2))-GC(I)*V(I,1,1)
1012 DV(I,1,2)=LX(I)*(V(I,1,1))-V(I,2,1))-RLU(I)*V(I,1,2)
1013 DV(I,1,3)=LY(I)*(V(I,1,1))-V(I+1,1,1))-RLU(I)*V(I,1,3)
DV(I,NY,1)=(LX(I)/CXY2(I))*DV(I,NYM1,2)
DO 1000 J=2,NYM1
1 DV(I,J,1)=CXY(I)*(V(I,J,2)+V(I-1,J,3)-V(I,J,3))-GC(I)*
&V(I,J,1)
2 DV(I,J,2)=LX(I)*(V(I,J,1))-V(I,J+1,1))-RLU(I)*V(I,J,2)
3 DV(I,J,3)=LY(I)*(V(I,J,1))-V(I+1,J,1))-RLU(I)*V(I,J,3)
1000 CONTINUE
DO 2000 J=2,NYM1

```



```

1101 DV(1,J,1)=CXY2(I)*(V(1,J-1,2)-V(1,J,2)-V(1,J,3))-GC(1)*V(1,J,1)
1102 DV(1,J,2)=LX(1)*(V(1,J,1)-V(1,J+1,1))-RLU(1)*V(1,J,2)
1103 DV(1,J,3)=LY(1)*(V(1,J,1)-V(2,J,1))-RLU(1)*V(1,J,3)
1901 DV(NX,J,1)=CXY2(I)*(V(NX,J-1,2)-V(NX,J,2)+V(NXM1,J,3))-GC(NX)*V(NX
      J,1)
1902 DV(NX,J,2)=LX(NX)*(V(NX,J,1)-V(NX,J+1,1))-RLU(NX)*V(NX,J,2)
1903 DV(NX,J,3)=0.
2000 RETURN
      DO 3000 I=2,NXM1
      DV(I,1,2)=LX(1)*(V(I,1,1)-V(I,2,1))-RLU(1)*V(I,1,2)
      DV(I,1,3)=LY(1)*(V(I,1,1)-V(I+1,1,1))-RLU(1)*V(I,1,3)
      DV(I,NY,2)=0.
      DV(I,NY,3)=LY(1)*(V(I,NY,1)-V(I+1,NY,1))-RLU(1)*V(I,NY,3)
      DO 3000 J=2,NYM1
      DV(I,J,2)=LX(1)*(V(I,J,1)-V(I,J+1,1))-RLU(1)*V(I,J,2)
      DV(I,J,3)=LY(1)*(V(I,J,1)-V(I+1,J,1))-RLU(1)*V(I,J,3)
      CONTINUE
      DO 4000 J=2,NYM1
      DV(1,J,2)=LX(1)*(V(1,J,1)-V(1,J+1,1))-RLU(1)*V(1,J,2)
      DV(1,J,3)=LY(1)*(V(1,J,1)-V(2,J,1))-RLU(1)*V(1,J,3)
      DV(NX,J,2)=LX(1)*(V(NX,J,1)-V(NX,J+1,1))-RLU(1)*V(NX,J,2)
      DV(NX,J,3)=0.
      CONTINUE
      RETUPN
      END
4000

```

```

SUBROUTINE RKUTTA(V)
REAL CRK(4)/C.0,0.5,0.5,1.0/
REAL F16TH/0.1666667/
COMMON/TDT/T,DT,DT1,DT2,DT3
COMMON/SIZE/NX,NY,NT,IIS1,IIS2
COMMON/PGMC/IWRT,IDRW,IEN,IANL,ISAMP,IDFT,NWV,JDET,ITYPE
COMMON/STF/II,NARB,NARE
REAL DV(25,25,3)
COMMON/DERIV/DV
DIMENSION V(25,25,3),VC(25,25,3),AK(25,25,3,4)
CALL VEGUAS(V)
DO 1 K=1,3
DO 1 J=1,NY
DO 1 I=1,NX
1 AK(I,J,K,1)=DT*DV(I,J,K)
DO 3 L=2,4
LM1=L-1
CRKD=CRK(L)

```

```

DO 2 K=1,3
DO 2 J=1,NX
DO 2 I=1,NX
2 VC(I,J,K)=V(I,J,K)+CRKD*AK(I,J,K,LM1)
DO 3 K=1,3
DO 3 J=1,NY
DO 3 I=1,NX
3 AK(I,J,K,L)=DT*DV(I,J,K)
DO 4 K=1,3
DO 4 J=1,NY
DO 4 I=1,NX
4 V(I,J,K)=V(I,J,K)+(AK(I,J,K,1)+2.*(AK(I,J,K,2)+AK(I,J,K,3))+AK(I,J,K,4))*F16TH
5 RETURN
END

```

```

FUNCTION ENERGY(V)
REAL LX(25),LXI(25),LY(25),LYI(25),CXY(25),CXYI(25),CXY2(25),
&CXY2I(25),CXY4(25),CXY4I(25),RLU(25),GC(25)
COMMON/EL/LX,LXI,LY,LYI,CXY,CXYI,CXY2,CXY2I,CXY4,CXY4I,RLU,GC
COMMON/IDT/T,DT,DT1,DT2,DT3
COMMON/SIZE/NX,NY,NT,IIS1,IIS2
COMMON/PGMC/IWRT,IDRW,IEN,IANL,ISAMP,IDFT,NWV,JDET,ITYPE
DIMENSION V(25,25,3)
NXM1=NX-1
NYM1=NY-1
EN=0.
DO 1000 I=2,NXM1
EN=EN+CXY2I(I)*V(I,1,1)**2
1011 EN=EN+LXI(I)*V(I,1,2)**2
1012 EN=EN+LXI(I)*V(I,1,3)**2
1013 EN=EN+CXY2I(I)*V(I,NY,1)**2
1091 EN=EN
1092 EN=EN+LYI(I)*V(I,NY,3)**2
1093 EN=EN+LXI(I)*V(I,J,1)**2
DO 1000 J=2,NYM1
EN=EN+CXYI(I)*V(I,J,1)**2
1 EN=EN+LXI(I)*V(I,J,2)**2
2 EN=EN+LXI(I)*V(I,J,3)**2
3 CONTINUE
DO 2000 J=2,NYM1
EN=EN+CXY2I(I)*V(I,J,1)**2
1101 EN=EN+LXI(I)*V(I,J,2)**2
1102 EN=EN+LXI(I)*V(I,J,3)**2
1103 EN=EN+CXY2I(NX)*V(NX,J,1)**2
1901 EN=EN+CXY2I(NX)*V(NX,J,1)**2

```

```

1902 EN=EN+LXI(NX)*V(NX,J,2)**2
1903 EN=EN
2000 CONTINUE
1111 EN=EN+CXI(1)*V(1,1,1)**2
1112 EN=EN+LXI(1)*V(1,1,2)**2
1113 EN=EN+LYI(1)*V(1,1,3)**2
1191 EN=EN+CXI(1)*V(1,NY,1)**2
1192 EN=EN+LYI(1)*V(1,NY,3)**2
1193 EN=EN+CXI(1)*V(1,NY,2)**2
1911 EN=EN+LXI(NX)*V(NX,1,1)**2
1912 EN=EN+LXI(NX)*V(NX,1,2)**2
1913 EN=EN+CXI(NX)*V(NX,NY,1)**2
1991 EN=EN
1992 EN=EN
1993 ENERGY=EN
      RETURN
      END

```

```

C SUBROUTINE DRAW3D(V)
C DRAW3D USES SUBROUTINE DRAW TO MAKE A CONTOUR MAP SHOWING THE PRESSURE
C DISTRIBUTION FOR TWO-DIMENSIONAL PROBLEMS. FOR THREE-DIMENSIONAL PROBLEMS
C TWO DIMENSIONAL LAYER PLOTS MUST BE USED.

      REAL*8 ITI(10,12), IT(12)
      REAL*4 LA(100)
      INTEGER NG/1/
      COMMON/DT/DT,DT1,DT2,DT3
      COMMON/SIZE/NX,NY,NT,ISI,IS2
      COMMON/PGMC/IWRT,IDRW,IEN,IANL,ISAMP,IDFT,NWV,JDET,ITYPE
      COMMON/SCALE/EX,EY,SLCPE,FACT
      COMMON/TITLE/ITI,LA
      COMMON/DRW/IU,IR,I XL,IYL,IW,IH,IG
      DIMENSION V(25,25,3),X(25),Y(25),7(25),TEMP(25,25)
      MC=1
      DO 1 I=1,12
        IT(I)=ITI(NG,I)
        VMAX=0.0
        DO 11 J=1,NY
          DO 11 I=1,NX
            TEMP(I,J)=V(I,J,1)
            VABS=ABS(TEMP(I,J))
            IF(VABS.GT.VMAX) VMAX=VABS
          IF(VABS.GT.VMAX) VMAX=VABS
        11 CONTINUE
        IF(VMAX.NE.1.0) GO TO 12
        GO TO 14

```

```

12 DEL=1.0/VMAX
DO 13 J=1,NY
DO 13 I=1,NX
13 TEMP(I,J)=DEL*TEMP(I,J)
14 WRITE(6,20)DEL
DO 3 I=1,NX
FIM1=I-1,NY
DO 2 M=1,NY
FMM1=M-1
Z(M)=FACT*TEMP(I,M)
X(M)=FMM1+FIM1*SLOPE
2 Y(M)=FIM1+Z(M)
IF(I.GT.1)MC=2
3 CALL DRAW(NY,X,Y,MC,0,LA(I),IT,EX,EY,IU,IR,IXL,IYL,IW,IH,IG,LA(I))
DO 5 J=1,NY
FJM1=J-1,NX
DO 4 M=1,NX
FMM1=M-1
Z(M)=FACT*TEMP(M,J)
X(M)=FJM1+FMM1*SLOPE
4 Y(M)=Z(M)+FMM1
IF(J.EQ.NY)MC=3
5 CALL DRAW(NX,X,Y,MC,0,LA(J),IT,EX,EY,IU,IR,IXL,IYL,IW,IH,IG,LA(J))
NG=NG+1
20 FORMAT(//5H DEL=,F10.4///)
RETURN
END

```

```

C GEN
SUBROUTINE GEN(V)
IS THE SOURCE SUBROUTINE
COMMON/DT1,T,DT,DT1,DT2,DT3
COMMON/SIZE/NX,NY,NT,IIS1,IIS2
COMMON/PGMC/IWRT,IDRW,IEN,IANL,ISAMP,IDFT,NWV
REAL PI2/6.28318/F/ 4.0/
DIMENSION V(25,25,3)
FUN=75.0
PI2T=6.283185*T
V( 2, 2, 1)=1.0*COS(PI2T*FUN)
RETURN
END

```

```

SUBROUTINE STORE(V,DA)
COMPLEX*8DA(10,16)
COMMON/STF/II,NY,NT,II S1,II S2
COMMON/NARB,NARE
COMMON/PGMC/IWRT,I DRW,IEN,IANL,ISAMP,IDFT,NWV,JDET,ITYPE
DIMENSION V(25,25,3)
INDEX=ICD(II,ISAMP)
IF(INDEX.EQ.0)INDEX=ISAMP
DO 1 I=1,10
1 DA(I,INDEX)=V(I,JDET,I)
RETURN
END

```

```

SUBROUTINE CALG1(A,FCRM)
COMPLEX*8A(16)
COMMON/RARG/B(16),ARG(16),N1
COMMON/PGMC/IWRT,I DRW,IEN,IANL,ISAMP,IDFT,NWV,JDET,ITYPE
DIMENSION IM1(16)
DIMENSION FORM(40)
DO 1 II=1,ISAMP
IM1(II)=I1-I
B(II)=CABS(A(II))
AR=REAL(A(II))
AI=AIMAG(A(II))
IF(AR.EQ.0.0)AR=1.0E-06
ARG(II)=57.29578*ATAN2(AI,AR)
1 WRITE(6,FCRM)(IM1(II),A(II),B(II),ARG(II),II=1,ISAMP)
RETURN
END

```

```

SUBROUTINE FFT(DA,FORMD,FORMF)
COMPLEX*8DA(10,16),PI2J/(0.0,6.28318)/
COMPLEX*8DU(16)
INTEGER M(3)/4,0,0/
COMMON/BARG/B(16),ARG(16),N1
COMMON/PGMC/IWRT,I DRW,IEN,IANL,ISAMP,IDFT,NWV,JDET,ITYPE
COMMON/STF/II,NARB,NARE
DIMENSION FCRMD(40),FCRMF(40)
DO 2 I=1,10
DO 1 J=1,ISAMP
1 DU(J)=DA(I,J)
2

```



```

CALL CALG1(DU,FORMD)
CALL HARM(DU,M,INV,S,-1,IFERR)
CALL CALG1(DU,FORMF)
DO 2 J=1,ISAMP
  2 DA(I,J)=B(J)+(0.0,1.0)*ARG(J)
RETURN
END

SUBROUTINE DAMAX(DA,FMAX)
COMPLEX DA(10,16)
COMMON/PGMC/IWRT,IDRW,IEN,IANL,ISAMP,IDFT,NWV,JDET,ITYPE
FMAX=0.0
DO 1 I=1,10
DO 1 J=1,ISAMP
TEMP=REAL(DA(I,J))
IF(TEMP.GT.FMAX)FMAX=TEMP
1 CONTINUE
FMAX=FMAX/15.
RETURN
END

C GENERAL 3-DIMENSIONAL OVERLAY PROGRAM FOR INTEGRATION OF THE WAVE EQUATION
C USING RUNGE-KUTTA 4-TH ORDER INTEGRATION OF THE CONTINUOUS DISCRETE SYSTEM
C THE PRESSURE AND 3 VELOCITY COMPONENTS ARE STORED IN THE LARGE ARRAY
C V(I,J,KLM) WITH THE TIME DERIVATIVE IN DV(IJKL). ALL CALCULATIONS ARE ACTUALLY
C PERFORMED ON THE LARGE ARRAY V(I,J,K,L,M) WHICH IS INDEXED AS A SINGLE
C SUBSCRIPT ARRAY WITH INDEX IJKLM. OVERLAY ARRAYS OF BOUNDARY VALUES ARE STORED
C ON MAGNETIC DISC FILE 8 WITH 1500000 60 BIT WORDS WHEN RUN ON A CDC6500.
C ANY OF THE OTHER COMMON INTEGRATION SCHEMES THAT EVALUATE THE TIME DERIVATIVE
C DV(I)/DT AS A FUNCTION OF THE PRESENT VALUES OF THE VARIABLES V(I) MAY BE USED
C IN PLACE OF RKUT SUCH AS RUNGE-KUTTA-ADAMS-MOULTON,ADAMS-BASHFORTH,ETC.
*BNNEW(8,600),CNEW(8,600),BOLD(8,600),COLD(8,600),IIN(4),INV(4),M(3),
*INTEGER ARRAYS,OK
COMMON/MEDIUM/VMDV(400),BMD(400),BMDI(400),FX(400),
*FY(400),FZ(400),HX(400),HZ(400),ZSIDE(400),YSIDE(400),
*ZFRT(100),ZBACK(100),YBACK(100),ZTOP(100),YTOP(100),ZBOT(100),
COMMON/SIZE/N1M1,N1,N1P1,N2M1,N2,N2P1,N3M1,N3,N3P1,N12,N13,N14,N15,N17,LT
*,N23,I1,I2,I3,I4,I5,I6,I7,I8,I9,I10,I11,I12,I13,I14,I15,I16,I17,LT
*,N132,N133,N142,N143
COMMON/SIG/NXY7S,NFREQ
COMMON/TDIT/DT(20),TS,DX,DY,DZ
EQUIVALENCE(NX,N1),(NY,N2),(NZ,N3)
DATA NW1/10/,NW2/5/,NW3/10/,M(1)/3/,M(2)/0/,M(3)/J/,IWRT/50/
DATA IEN/1/,II/O/,KK/O/,NSAMP/R/,IIMIN/60/,INDEX/O/,IDTS/20/
DATA ICW/1/,IBW/1/,IIN(1)/O/,IIN(2)/1/,IIN(3)/2/,IIN(4)/3/,IPW/1/

```



```

DATA ISHS/1/,ISHB/1/,ISHBK/1/,ISHF/1/,ISHT/1/,IS/1/,IT/O/,IB/O/
DATA RANGE/O.O/,RANGEM/100.O/,STABLE/10.O/,ERROR/O.O1/,UAVE/O.O.O/
DATA DEPTH/O.O/,DEPTHM/100.O/
C
C
C
V(IJKLM)=V(I+(J-1)*N1+(K-1)*N12+(L-1)*N13+(M-1)*N14)
READ(5,1C)NX,NY,NZ,NT,NXS,NYS,NZS,NFREQ
WRITE(6,290)
WRITE(6,40)NX,NY,NZ,NT,NXS,NYS,NZS,NFREQ
READ(5,150)DX,DY,DZ
READ(5,20)(DT(I),I=1,20)
WRITE(6,160)DX,DY,DZ
WRITE(6,360)(DT(I),I=1,20)
READ(5,150)FUN,(AMP(I),I=1,NFREQ)
WRITE(6,180)FUN,(AMP(I),I=1,NFREQ)
N1M1=N1-1
N2M1=N2-1
N3M1=N3-1
N1P1=N1+1
N2P1=N2+1
N3P1=N3+1
N12=N1*N2
N13=N12*N3
N14=N13*N4
N15=N14*N6
N132=2*N13
N133=3*N13
N142=2*N14
N23=N2*N3
N24=N23*N4
I1=-N1-N12
I2=-N13-N12-N1
I3=1-N12-N1
I4=1-N1-N12+N13
I5=N1M1-N1-N12+N13
I6=-N2-N23
I7=-N12+N132
I8=-N1+N12-N13
I9=2-N1-N12-N13
I10=-N1+N132
I11=N12-N1
I12=N13-N12
I13=-N12+N133
N1N3=N1*N3
KLMT=24*N3
NXYZS=NXS+(NYS-1)*N1+(NZS-1)*N12
T=0.0
WRITE(6,60)N1,N2,N3,N12,N13,N14,N15

```

```

MV3DC150
MV3D0160
MV3DC170
MV3DC180
MV3D0190
MV3D0200
MV3DC210
MV3DC220
MV3DC230
MV3DC240
MV3DC250
MV3D0260
MV3DC270
MV3D0280
MV3DC290
MV3D0300
MV3D0310
MV3D0320
MV3D0330
MV3D0340
MV3D0350
MV3DC360
MV3D0370
MV3D0380
MV3D0390
MV3D0400
MV3DC410
MV3D0420
MV3D0430
MV3D0440
MV3D0450
MV3D0460
MV3D0470
MV3D0480
MV3DC490
MV3D0500
MV3DC510
MV3D0520
MV3D0530
MV3D0540
MV3D0550
MV3D0560
MV3D0570
MV3D0580
MV3DC590
MV3D0600
MV3D0610
MV3D0620

```

```

ARRAYS=N15+N14+12*N1*N3+6*N1*N2
WRITE(6,90)ARRAYS
VOL2=0.5*DX*DY*DZ
DO 1 K=1,N3
KI=IK
DO 1 I=1,N1
IK=KI+I
VMD(IK)=1.0E+03
VMDV(IK)=VOL2*VMD(IK)
RMD(IK)=2.25E+09
BMDI(IK)=VOL2/BMD(IK)
HX(IK)=1.0/(DX*VMD(IK))
HY(IK)=1.0/(DY*VMD(IK))
HZ(IK)=1.0/(DZ*VMD(IK))
FX(IK)=BMD(IK)/DX
FY(IK)=BMD(IK)/DY
FZ(IK)=BMD(IK)/DZ
ZSIDE(IK)=SQRT(VMD(IK)*BMD(IK))
IF(ISHS.EQ.0)ZSIDE(IK)=0.0
YSIDE(IK)=HY(IK)
1 CONTINUE
DO 2 KLM=1,KLMT
DO 2 J=1,N2
DO 2 I=1,N1
2 V(I,J,KLM)=0.0
DO 3 IJKL=1,N14
DO 3 DV(IJKL)=0.0
DO 5 K=1,N3
JK=-N2+K*N2
N1M1K=-1+K*N1
OK=1-N1+K*N1
KJ=JK
DO 5 J=1,N2
JK=KJ+J
ZFRIT(JK)=ZSIDE(N1M1K)
IF(ISHF.EQ.0)ZFRIT(JK)=0.0
ZBACK(JK)=ZSIDE(OK)
IF(ISHBK.EQ.0)ZBACK(JK)=0.0
YBACK(JK)=HX(OK)
5 CONTINUE
DO 6 J=1,N2
IJ=-N1+J*N1
JI=IJ
DO 6 I=1,N1
IJ=JI+I
INM=N1N3-2*N1+I
ZTOP(IJ)=ZSIDE(I)

```

```

VV3D00630
VV3D00640
VV3D00650
VV3D00660
VV3D00670
VV3D00680
VV3D00690
VV3D00700
VV3D00710
VV3D00720
VV3D00730
VV3D00740
VV3D00750
VV3D00760
VV3D00770
VV3D00780
VV3D00790
VV3D00800
VV3D00810
VV3D00820
VV3D00830
VV3D00840
VV3D00850
VV3D00860
VV3D00870
VV3D00880
VV3D00890
VV3D00900
VV3D00910
VV3D00920
VV3D00930
VV3D00940
VV3D00950
VV3D00960
VV3D00970
VV3D00980
VV3D00990
VV3D1000
VV3D1010
VV3D1020
VV3D1030
VV3D1040
VV3D1050
VV3D1060
VV3D1070
VV3D1080
VV3D1090
VV3D1100

```

WV3D11110
 WV3D11120
 WV3D11130
 WV3D11140
 WV3D11150
 WV3D11160
 WV3D11170
 WV3D11180
 WV3D11190
 WV3D1200
 WV3D1210
 WV3D1220
 WV3D1230
 WV3D1240
 WV3D1250
 WV3D1260
 WV3D1270
 WV3D1280
 WV3D1290
 WV3D1300
 WV3D1310
 WV3D1320
 WV3D1330
 WV3D1340
 WV3D1350
 WV3D1360
 WV3D1370
 WV3D1380
 WV3D1390
 WV3D1400
 WV3D1410
 WV3D1420
 WV3D1430
 WV3D1440
 WV3D1450
 WV3D1460
 WV3D1470
 WV3D1480
 WV3D1490
 WV3D1500
 WV3D1510
 WV3D1520
 WV3D1530
 WV3D1540
 WV3D1550
 WV3D1560
 WV3D1570
 WV3D1580

```

IF (ISHT.EQ.0) ZTOP(IJ)=0.0
YTOP(IJ)=HZ(I)
ZBOT(IJ)=ZSIDE(INM)
IF (ISHB.EQ.0) ZBOT(IJ)=0.0
6 CONTINUE
IF (IPW.EQ.0) GO TO 8
WRITE(6,130)
WRITE(6,170) (VMD (IK), IK=1,N1N3)
WRITE(6,170) (VMDV(IK), IK=1,N1N3)
WRITE(6,140)
WRITE(6,170) (BMD (IK), IK=1,N1N3)
WRITE(6,180)
WRITE(6,170) (BMDI(IK), IK=1,N1N3)
WRITE(6,340)
WRITE(6,170) (ZSIDE(IK), IK=1,N1N3)
WRITE(6,350)
WRITE(6,170) (YSIDE(IK), IK=1,N1N3)
WRITE(6,210)
WRITE(6,170) (HX(IK), IK=1,N1N3)
WRITE(6,220)
WRITE(6,170) (HY(IK), IK=1,N1N3)
WRITE(6,230)
WRITE(6,170) (HZ(IK), IK=1,N1N3)
WRITE(6,240)
WRITE(6,170) (FX(IK), IK=1,N1N3)
WRITE(6,250)
WRITE(6,170) (FY(IK), IK=1,N1N3)
WRITE(6,260)
WRITE(6,170) (FZ(IK), IK=1,N1N3)
WRITE(6,270)
WRITE(6,170) (ZFRT(JK), JK=1,N23)
WRITE(6,280)
WRITE(6,170) (ZBACK(JK), JK=1,N23)
WRITE(6,300)
WRITE(6,170) (YBACK(JK), JK=1,N23)
WRITE(6,320)
WRITE(6,170) (ZTCP (IJ), IJ=1,N12)
WRITE(6,330)
WRITE(6,170) (YTOP (IJ), IJ=1,N12)
WRITE(6,310)
WRITE(6,170) (ZBOT (IJ), IJ=1,N12)
8 CONTINUE
I7=1
IX=1
CONTINUE
II=II+1
IIM1=II-1
  
```

```

IF(II.LT.21)LT=II
T=T+DT(LT)
IF(IS.EQ.1)CALL SOURCE(V,AMP,FUN)
IF(IB.EQ.1)CALL BACK(V,BOLD,FUN)
IF(IT.EQ.1)CALL TOP(V,COLD,FUN)
CALL RKUT(V,DV)
C AVERAGE ENERGY CALCULATION
ENERGY=U(V)
UAVE=UAVE+(ENERGY-UAVE)*DT(LT)/T
IF(II.GT.IIMIN)STABLE=ABS(UAVE-SUAVE)
ERROR=0.1*UAVE
SUAVE=UAVE
IF(MOD(II,IPLT).EQ.0)CALL DRAW3D(V)
IF(MOD(IIM1,IWRT))14,13,14
13 WRITE(6,400)T
14 WRITE(6,100)((I,J,KLM,V(I,J,KLM),KLM=1,NW3),J=1,NW2),I=1,NW1)
15 WRITE(6,50)ENERGY,II,T,UAVE,STABLE
IF(IIM1.EQ.0)GO TO 16
PEC=(ENERGY-ELAST)*100.0/ELAST
WRITE(6,110)PEC
16 CONTINUE
ELAST=ENERGY
IF(STABLE.LT.ERROR)GO TO 17
IISS=II+1
TSS=T+DT(LT)
IF(II.GT.NT)STOP
GO TO 9
17 IF(II.EQ.IISS)GO TO 18
IF(II-IINS)9,18,18
18 IINS=II+IDTS
INDEX=INDEX+1
DO 19 L=1,4
JKLE=I6+L*N23
LK=JKLE
KL=-N3+L*N3
LK1=KL
DO 19 K=2,N3M1
JKLE=LK+K*N2
KJ=JKLE
KL=LK1+K
DO 19 J=2,N2M1
JKLE=2*(KJ+J)
JKLO=JKLE-1
BNEW(INDEX,JKLE)=0.0
19 BNEW(INDEX,JKLO)=V(N1M1,J,KL)
DO 21 L=1,4
IJLE=II+L*N12

```

WV3D1590
WV3D160C
WV3D1610
WV3D1620
WV3D163C
WV3D1640

WV3D1650
WV3D166C
WV3D1670
WV3D168C
WV3D1690
WV3D1700
WV3D1710
WV3D172C
WV3D173C
WV3D1740
WV3D1750
WV3D1760
WV3D1770
WV3D178C
WV3D1790
WV3D180C
WV3D1810
WV3D1820
WV3D1830
WV3D184C
WV3D185C
WV3D1860
WV3D1870
WV3D1880
WV3D1890
WV3D1900
WV3D191C
WV3D192C
WV3D1930
WV3D1940
WV3D1950
WV3D1960
WV3D1970
WV3D1980
WV3D1990
WV3D2000
WV3D2010
WV3D2020
WV3D2030
WV3D2040
WV3D2050

HV3D206C
HV3D207C
HV3D208C
HV3D209C
HV3D210C
HV3D211C
HV3D212C
HV3D213C
HV3D214C
HV3D215C
HV3D216C
HV3D217C
HV3D218C
HV3D219C
HV3D220C
HV3D221C
HV3D222C
HV3D223C
HV3D224C
HV3D225C
HV3D226C
HV3D227C
HV3D228C
HV3D229C
HV3D230C
HV3D231C
HV3D232C
HV3D233C
HV3D234C
HV3D235C
HV3D236C
HV3D237C
HV3D238C
HV3D239C
HV3D240C
HV3D241C
HV3D242C
HV3D243C
HV3D244C
HV3D245C
HV3D246C
HV3D247C
HV3D248C
HV3D249C
HV3D250C
HV3D251C
HV3D252C
HV3D253C

```

LJ=IJLE
NML=-1+L*N3
DO 21 J=2,N2M1
IJLE=LJ+J*N1
JI=IJLE
DO 21 I=2,N1M1
IJLE=2*(JI+I)
IJLO=IJLE-1
CNEW(INDEX,IJLE)=C.O
CNEW(INDEX,IJLO)=V(I,J,NML)
21 WRITE(6,1C)INDEX,II
IF(INDEX-NSAMP)9,23,23
23 IF(ICW.EQ.1)WRITE(6,370)IZ,IX
DO 27 L=1,4
IJLE=II+L*N12
LJ=IJLE
DO 27 J=2,N2M1
IJLE=LJ+J*N1
JI=IJLE
DO 27 I=2,N1M1
IJLE=2*(JI+I)
IJLO=IJLE-1
DO 24 INDEX=1,NSAMP
IND2=2*INDEX
IND2M=IND2-1
A(IND2)=C.O
24 A(IND2M)=CNEW(INDEX,IJLO)
CALL HARM(A,M,INV,S,-1,IFERR)
DO 25 INDEX=1,NFREQ
IND2=2*INDEX
IND2M=IND2-1
CNEW(INDEX,IJLE)=ATAN2(A(IND2),A(IND2M))
25 CNEW(INDEX,IJLO)=2.O*SORT(A(IND2)*2+A(IND2M)*2)
WRITE(8)(CNEW(IN,IJLO),CNEW(IN,IJLE),IN=2,NFREQ)
IF(ICW.EQ.1)WRITE(6,200)(IIN(IN),I,J,N3M1,L,CNEW(IN,IJLO),CNEW(IN,
*IJLE),IN=1,NFREQ)
27 CONTINUE
IF(1BW.EQ.1)WRITE(6,380)IZ,IX
DO 34 L=1,4
JKLE=I6+L*N23
LK=JKLE
DO 34 K=2,N3M1
JKLE=LK+K*N2
KJ=JKLE
DO 34 J=2,N2M1
JKLE=2*(KJ+J)
JKLO=JKLE-1
DO 31 INDEX=1,NSAMP

```



```

IND2=2*INDEX
IND2M=IND2-1
A(IND2)=0.0
31 A(IND2M)=8NEW(INDEX,JKLC)
DO 32 CALL HARM(A,M,INV,S,-1,IFERR)
DO 32 INDEX=1,NFREQ
IND2=2*INDEX
IND2M=IND2-1
BNEW(INDEX,JKLE)=ATAN2(A(IND2),A(IND2M))
32 BNEW(INDEX,JKLO)=2.0*SQRT(A(IND2)**2+A(IND2M)**2)
WRITE(8)(BNEW(IN,JKLO),BNEW(IN,JKLE),IN=2,NFREQ)
IF(IBW.EQ.1)WRITE(6,200)(IIN(IN),N1M1,J,K,L,BNEW(IN,JKLO),BNEW(IN,JKLE),IN=1,NFREQ)
*JKLE),IN=1,NFREQ)
34 CONTINUE
DO 41 KLM=1,KLMT
DO 41 J=1,N2
DO 41 I=1,N1
41 V(I,J,KLM)=0.0
DO 42 IJKL=1,N14
42 DV(IJKL)=0.0
IF(IX.EQ.1)AND(IZ.EQ.1)GO TO 51
IF(IZ.EQ.1)GO TO 53
51 CONTINUE
DO 52 L=1,4
IJLE=I1+L*N12
IJ=IJLE
DO 52 J=2,N2M1
IJLE=LJ+J*N1
JI=IJLE
DO 52 I=2,N1M1
IJLE=2*(JI+I)
IJLO=IJLE-1
READ(8)(COLD(IN,IJLO),COLD(IN,IJLE),IN=2,NFREQ)
52 CONTINUE
53 CONTINUE
IF(IX.EQ.1)GO TO 69
DO 62 L=1,4
JKLE=I6+L*N23
LK=JKLE
DO 62 K=2,N3M1
JKLE=LK+K*N2
KJ=JKLE
DO 62 J=2,N2M1
JKLE=2*(KJ+J)
JKLO=JKLE-1
READ(8)(BOLD(IN,JKLO),BOLD(IN,JKLE),IN=2,NFREQ)
62 CONTINUE

```

WV3D2540
 WV3D2550
 WV3D2560
 WV3D2570
 WV3D2580
 WV3D2590
 WV3D2600
 WV3D2610
 WV3D2620
 WV3D2630
 WV3D2640
 WV3D2650
 WV3D2660
 WV3D2670
 WV3D2680
 WV3D2690
 WV3D2700
 WV3D2710
 WV3D2720
 WV3D2730
 WV3D2740
 WV3D2750
 WV3D2760
 WV3D2770
 WV3D2780
 WV3D2790
 WV3D2800
 WV3D2810
 WV3D2820
 WV3D2830
 WV3D2840
 WV3D2850
 WV3D2860
 WV3D2870
 WV3D2880
 WV3D2890
 WV3D2900
 WV3D2910
 WV3D2920
 WV3D2930
 WV3D2940
 WV3D2950
 WV3D2960
 WV3D2970
 WV3D2980
 WV3D2990
 WV3D3000
 WV3D3010


```

69 CONTINUE
UAVE=0.0
INDEX=0
IF(1B.EQ.1)RANGE=RANGE+FLOAT(N1)*DX
IF(RANGE.GT.RANGEM)STCP
IF(1T.EQ.1)DEPTH=DEPTH+FLOAT(N3)*DZ
IF(DEPTH.GT.DEPTHM)STCP
STABLE=10.0
II=0
ERROR=0.01
T=0.0
TS12=TS
IT=1
IS=0
PAUSE 1
GO TO 9
10 FORMAT(8I10)
20 FORMAT(CP10E8.1)
30 FORMAT(8A10)
40 FORMAT(/4H NX=,I2,4H NY=,I2,4H NZ=,I2,7H NREQ=,I1)
* I2,5H NZS=,I2,7H NREQ=,I1)
50 FORMAT(8H ENERGY=,E14.7,14H AT TIME STEP ,I3,7H, TIME=,E14.7,
* 6H SEC.,,13H AVE. ENERGY=,E14.7,9H, STABLE=,E14.7)
60 FORMAT(/4H N1=,I2,4H N2=,I2,4H N3=,I2,
* I5,5H N13=,I5,5H N14=,I5,5H N15=,I5)
70 FORMAT(/10H VMDV(I,K))
80 FORMAT(/10H BMDI(I,K))
90 FORMAT(/24H STORAGE SIZE OF ARRAYS=,I10)
100 FORMAT(/7(1X,I2,1H*,I1,1H*,I2,1X,OP1E10.3)))
110 FORMAT(/49H PERCENT ENERGY CHANGE RELATIVE TO LAST ANALYSIS=
* OP1E15.8)
130 FORMAT(9H VMD(I,K))
140 FORMAT(9H BMD(I,K))
150 FORMAT(CP8E10.3)
160 FORMAT(/4H DX=,OP1E10.3,4H DY=,OP1E10.3,4H DZ=,OP1E10.3)
170 FORMAT(1X,OP13E10.3)
180 FORMAT(/19H FUNDAMENTAL FREQ.=,E10.3,29H HZ., HARMONIC AMPLITUDES
* ARE/(8E10.3))
200 FORMAT(3(2X,I1,2X,I1,2X,I1,2X,I1,1X,E10.3,1H,E10.3))
210 FORMAT(/8H HX(I,K))
220 FORMAT(/8H HY(I,K))
230 FORMAT(/8H HZ(I,K))
240 FORMAT(/8H FX(I,K))
250 FORMAT(/8H FY(I,K))
260 FORMAT(/8H FZ(I,K))
270 FORMAT(/10H ZFRT(J,K))
280 FORMAT(/11H ZBACK(J,K))
290 FORMAT(1H1)

```

```

HV3D 3020
HV3D 3030
HV3D 3040
HV3D 3050
HV3D 3060
HV3D 3070
HV3D 3080
HV3D 3090
HV3D 3100
HV3D 3110
HV3D 3120
HV3D 3130
HV3D 3140
HV3D 3150
HV3D 3160
HV3D 3170
HV3D 3180
HV3D 3190
HV3D 3200
HV3D 3210
HV3D 3220
HV3D 3230
HV3D 3240
HV3D 3250
HV3D 3260
HV3D 3270
HV3D 3280
HV3D 3290
HV3D 3300
HV3D 3310
HV3D 3320
HV3D 3330
HV3D 3340
HV3D 3350
HV3D 3360
HV3D 3370
HV3D 3380
HV3D 3390
HV3D 3400
HV3D 3410
HV3D 3420
HV3D 3430
HV3D 3440
HV3D 3450
HV3D 3460
HV3D 3470
HV3D 3480
HV3D 3490

```

```

300 FORMAT(/11H YBACK(J,K))
310 FORMAT(/10H ZBOT(I,J))
320 FORMAT(/10H ZTOP(I,J))
330 FORMAT(/10H YTOP(I,J))
340 FORMAT(/11H ZSIDE(I,K))
350 FORMAT(/11H YSIDE(I,K))
360 FORMAT(/4H DT=(UPIOE10.3)
370 *LOCK I7=I5,5H,IX=I5/
    *114H HARM I J K L AMP(IJKL) PHASE(IJKL) HARM I J K L AMP(IJKL)
    *L) PHASE(IJKL) HARM I J K L AMP(IJKL) PHASE(IJKL)
380 FORMAT(/64H EQUIVALENT AMPLITUDE AND PHASE OF TOP MESHPOINTS FOR
    *RLOCK I7=I5,5H,IX=I5/
    *114H HARM I J K L AMP(IJKL) PHASE(IJKL) HARM I J K L AMP(IJKL)
    *L) PHASE(IJKL) HARM I J K L AMP(IJKL) PHASE(IJKL)
390 FORMAT(/23H AVERAGE STORED ENERGY=,E15.8)
400 FORMAT(/38H PRESSURE AT LOCATION I*J*K AT TIME T=,E15.8,5H SEC.)
    END

```

```

WV3D3500
WV3D3510
WV3D3520
WV3D3530
WV3D3540
WV3D3550
WV3D3560
WV3D3570
WV3D3580
WV3D3590
WV3D3600
WV3D3610
WV3D3620
WV3D3630
WV3D3640
WV3D3650
WV3D3660
WV3D3670

```

```

SUBROUTINE REQ(V,DV)
C REQ IS THE FIRST ORDER EQUATIONS EQUIVALENT TO THE WAVE EQUATION IN CONTINUOUS
C DISCRETE FORM.
    DIMENSION V(1),DV(1)
    INTEGER CJK1,CJK2,OPJK1,OPJK12,CJK12,OJK22
    COMMON/SIZE/N1M1,N1,N1P1,N2M1,N2,N2P1,N3M1,N3,N3P1,N12,N13,N14,N15,LT
    *,N23,I1,I2,I3,I4,I5,I6,I7,I8,I9,I10,I11,I12,I13,I14,I15,I16,I17,LT
    *,N132,N133,N142,N143
    COMMON/TDT/T,DT(20),TS,DX,DY,DZ
    COMMON/MEDIUM/VMD(400),VMDV(400),BMD(400),BMDI(400),FX(400),
    *FY(400),FZ(400),HX(400),HY(400),HZ(400),ZSIDE(400),YSIDE(400),
    *ZFRIT(100),ZBACK(100),YBACK(100),ZTOP(100),YTOP(100),ZBOT(100)
    C
    C V(IJKLM)=V(I+(J-1)*N1+(K-1)*N12+(L-1)*N13+(M-1)*N14)
    C
    DO 10000
    KN12=K*N12
    IJK1=I1+KN12
    KJ1=IJK1
    IK=-N1+K*N1
    IK1=IK
    OJK1=I3+KN12
    KJ2=OJK1
    NJK1=-N12+KN12
    KJ3=NJK1
    NMJK2=I5+KN12
    K=2,N3M1
    REQ 0010
    REQ 0020
    REQ 0030
    REQ 0040
    REQ 0050
    REQ 0060
    REQ 0070
    REQ 0080
    REQ 0090
    REQ 0100
    REQ 0110
    REQ 0120
    REQ 0130
    REQ 0140
    REQ 0150
    REQ 0160
    REQ 0170
    REQ 0180
    REQ 0190
    REQ 0200
    REQ 0210
    REQ 0220
    REQ 0230

```

REO REO REO REO REO REO REO REO REO REO
 0240 0250 0260 0270 0280 0290 0300 0310 0320 0330
 0340 0350 0360 0370 0380 0390 0400 0410 0420 0430
 0440 0450 0460 0470 0480 0490 0500 0510 0520 0530
 0540 0550 0560 0570 0580 0590 0600 0610 0620 0630
 0640 0650 0660 0670 0680 0690 0700 0710

J=2 ,N2M1

I=2,N1M1

J=2,N2M1

I=2,N1M1

K=2,N3M1

```

KJ4=NMJK2
JK=-N2+K*N2
KJ5=JK
DO 10000
JN1=J*N1
IJK1=KJ1+JN1
JI1=IJK1
OJK1=KJ2+JN1
OJK2=OJK1+N13
OPJK1=CJK1+I
NJK1=KJ3+JN1
NMJK2=KJ4+JN1
JK=KJ5+J
DV(OJK2)=YBACK(JK)*(V(OJK1)-V(OPJK1))
DV(OJK1)=-ZBACK(JK)*DV(OJK2)
DV(NJK1)=ZFRT(JK)*DV(NMJK2)
DO 10000
IJK1=JI1+I
IJK2=IJK1+N13
IJK3=IJK2+N13
IJK4=IJK3+N13
IPJK2=IJK2-I
IPJK1=IJK1+I
IJMK3=IJK3-N1
IJPk1=IJK1+N1
IJKP1=IJK1+N12
IJKM4=IJK4-N12
IK=IK1+I
DV(IJK1)=FX(IK)*(V(IJK2)-V(IJK3))+FY(IK)*(V(IJMK3)-V(IJK3))+
*FZ(IK)*(V(IJMK4)-V(IJK4))
DV(IJK2)=HX(IK)*(V(IJK1)-V(IPJK1))
DV(IJK3)=HY(IK)*(V(IJK1)-V(IJPK1))
DV(IJK4)=HZ(IK)*(V(IJK1)-V(IJKP1))
CONTINUE
10000
DO 10001
IJ11=-N1+J*N1
JI1=IJ11
DO 10001
IJ11=JI1+I
IJ14=IJ11+N133
IJ21=IJ11+N12
IJN1=IJ12+IJ11
IJNM4=IJ13+IJN1
DV(IJ14)=YTOP(IJ11)*(V(IJ11)-V(IJ21))
DV(IJ11)=-7TCP(IJ11)*DV(IJ14)
DV(IJN1)=ZBOT(IJ11)*DV(IJNM4)
CONTINUE
10001
DO 10002

```

REQ 0720
 REQ 0730
 REQ 0740
 REQ 0750
 REQ 0760
 REQ 0770
 REQ 0780
 REQ 0790
 REQ 0800
 REQ 0810
 REQ 0820
 REQ 0830
 REQ 0840
 REQ 0850
 REQ 0860
 REQ 0870
 REQ 0880
 REQ 0890
 REQ 0900
 REQ 0910
 REQ 0920
 REQ 0930
 REQ 0940
 REQ 0950
 REQ 0960
 REQ 0970
 REQ 0980
 REQ 0990
 REQ 1000
 REQ 1010
 REQ 1020
 REQ 1030
 REQ 1040
 REQ 1050
 REQ 1060
 REQ 1070
 REQ 1080
 REQ 1090
 REQ 1100
 REQ 1110
 REQ 1120
 REQ 1130
 REQ 1140
 REQ 1150
 REQ 1160
 REQ 1170
 REQ 1180
 REQ 1190

I=2,N1M1

K=2,N3M1

J=2,N2M1

I=2,N1M1

```

10002
I1K1=-N12+K*N12
KI1=I1K1
IK=-N1+K*N1
KI3=IK
DO 10002
  I1K1=KI1+I
  INMK3=I10+I1K1
  I2K1=I1K1+N1
  I1K3=I1K1+N132
  IK=KI3+I
  DV(I1K3)=YSIDE(IK)*(V(I1K1))-V(I2K1))
  DV(I1K1)=-ZSIDE(IK)*DV(I1K3)
  DV(INK1)=ZSIDE(IK)*DV(INMK3)
CONTINUE
RETURN
ENTRY REQ
DO 20000
  KN12=K*N12
  IJK1=I1+KN12
  KJ1=IJK1
  IK=-N1+K*N1
  IK1=IK
  OJK1=I3+KN12
  KJ2=OJK1
  NJK1=-N12+KN12
  KJ3=NJK1
  NMJK2=I5+KN12
  KJ4=NMJK2
  JK=-N2+K*N2
  KJ5=JK
DO 20000
  JN1=J*N1
  IJK1=KJ1+JN1
  JI1=IJK1
  OJK1=KJ2+JN1
  OJK12=OJK1+N14
  OJK2=OJK1+N13
  OPJK12=OJK1+I+N14
  NJK1=KJ3+JN1
  NMJK2=KJ4+JN1
  JK=KJ5+J
  DV(OJK2)=YBACK(JK)*(V(OJK12))-V(OPJK12))
  DV(OJK1)=-ZBACK(JK)*DV(OJK2)
  DV(NJK1)=ZFRT(JK)*DV(NMJK2)
DO 20000
  IJK1=JI1+I
  IJK12=IJK1+N14

```



```

IJK2=IJK1+N13
IJK22=IJK2+N14
IJK3=IJK2+N13
IJK32=IJK3+N14
IJK4=IJK3+N13
IJK42=IJK4+N14
IMJK22=IJK22-I
IPJK12=IJK12+I
IJMK32=IJK32-N1
IJPk12=IJK12+N1
IJKP12=IJK12+N12
IJKM42=IJK42-N12
IK=IK1+I
DV(IJK1)=FX(IK)*(V(IMJK22)-V(IJK22))+FY(IK)*(V(IJMK32)-V(IJK32))+
*FZ(IK)*(V(IJMK42)-V(IJK42))
DV(IJK2)=HX(IK)*(V(IJK12)-V(IPJK12))
DV(IJK3)=HY(IK)*(V(IJK12)-V(IJPK12))
DV(IJK4)=HZ(IK)*(V(IJK12)-V(IJKP12))
CONTINUE
DO 2000 JN1=J*N1
IJ11=-N1+JN1
JI1=IJ11
DO 2000 IJ11=I
IJ11=JI1+I
IJ112=IJ11+N14
IJ14=IJ11+N133
IJ212=IJ112+N12
IJN1=I12+IJ11
IJNM4=I13+IJN1
DV(IJ14)=YTOP(IJ11)*(V(IJ112)-V(IJ212))
DV(IJ11)=-ZTOP(IJ11)*DV(IJ14)
DV(IJN1)=ZBOT(IJ11)*DV(IJNM4)
CONTINUE
DO 2000 I1K1=-N12+K*N12
KI1=I1K1
IK=-N1+K*N1
KI3=IK
DO 2000 I1K1=KI1+I
I1K12=I1K1+N14
I1K3=I1K1+N132
I2K12=I1K12+N1
INK1=I11+I1K1
INMK3=I1C+INK1
IK=KI3+I
DV(I1K3)=YSIDE(IK)*(V(I1K12)-V(I2K12))

```

REO REO REO REO REO
1680 1690 1700 1710 1720

```

20002      DV(I,K1)=-ZSIDE(IK)*DV(I,K3)
          DV(I,K1)=ZSIDE(IK)*DV(I,K3)
          CONTINUE
          RETURN
          END

```

[illegible]

```

SUBROUTINE RKUT(V,DV)
DIMENSION V(1),DV(1),CRK(4)
COMMON/MEDIUM/VMD(400),VMDV(400),BMD(400),BMCI(400),FX(400),
**FY(400),FZ(400),HX(400),HY(400),HZ(400),ZSIDE(400),YSIDE(400),
**ZFRT(100),ZBACK(100),YBACK(100),ZTOP(100),YTCP(100),ZBOT(100)
COMMON/SIZE/NIM1,N1,N1P1,N2M1,N2,N2P1,N3M1,N3,N3P1,N12,N13,N14,N15
**,N23,I1,I2,I3,I4,I5,I6,I7,I8,I9,I10,I11,I12,I13,I14,I15,I16,I17,I18
**,N132,N133,N142,N143
COMMON/TDT/T,DT(20),TS,DX,DY,DZ
DATA CRK(1)/0.0/,CRK(2)/0.5/,CRK(3)/0.5/,CRK(4)/1.0/,F16TH/0.166666
**66666666667/

```

$$V(IJKLM) = V(I + (J-1)*N1 + (K-1)*N12 + (L-1)*N13 + (M-1)*N14)$$

```

CALL REQ(V,DV)
DO 10001
  IJKL1=I2 +L*N13
  LK1=IJKL1
DO 10001
  IJKL1=LK1+K*N12
  KJ1=IJKL1
DO 10001
  IJKL1=KJ1+J*N1
  JI1=IJKL1
DO 10001
  IJKL1=JI1+I
  IJKL3=IJKL1+N142
  IJKLV(IJKL3)=DV(LT)*DV(IJKL1)
CONTINUE
DO 10003
  CRKD=CRK(M-2)
  MN14=M*N14
  MMN14=-N142+MN14
  IJKLM=-N14+MN14
  MLI=IJKLM
DO 10002
  IJKL1=I2 +L*N13
  LK1=IJKL1
DO 10002

```


RKUT0390
 RKUT0400
 RKUT0410
 RKUT0420
 RKUT0430
 RKUT0440
 RKUT0450
 RKUT0460
 RKUT0470
 RKUT0480
 RKUT0490
 RKUT0500
 RKUT0510
 RKUT0520
 RKUT0530
 RKUT0540
 RKUT0550
 RKUT0560
 RKUT0570
 RKUT0580
 RKUT0590
 RKUT0600
 RKUT0610
 RKUT0620
 RKUT0630
 RKUT0640
 RKUT0650
 RKUT0660
 RKUT0670
 RKUT0680
 RKUT0690
 RKUT0700
 RKUT0710
 RKUT0720
 RKUT0730
 RKUT0740
 RKUT0750
 RKUT0760
 RKUT0770
 RKUT0780
 RKUT0790
 RKUT0800
 RKUT0810
 RKUT0820
 RKUT0830
 RKUT0840

J=1,N2

I=1,N1

L=1,4

K=1,N3

J=1,N2

I=1,N1

L=1,4

K=1,N3

J=1,N2

I=1,N1

*F16TH
 V(IJKL1)=V(IJKL1)+(V(IJKL3)+2.0*(V(IJKL4)+V(IJKL5))+V(IJKL6))*

```

10002  IJKL1=LK1+K*N12
        KJ1=IJKL1
        DO 10002
          IJKL1=KJ1+J*N1
          JI1=IJKL1
          DO 10002
            IJKL1=JI1+I
            IJKL2=IJKL1+N14
            IJKLMM=IJKL1+MMN14
            V(IJKL2)=V(IJKL1)+CRKD*V(IJKLMM)
          CONTINUE
        CALL REQC(V,DV)
        DO 10003
          IJKL1=I2+L*N13
          LK1=IJKL1
          DO 10003
            IJKL1=LK1+K*N12
            KJ1=IJKL1
            DO 10003
              IJKL1=KJ1+J*N1
              JI1=IJKL1
              DO 10003
                IJKL1=JI1+I
                IJKLM=MI1+IJKL1
                V(IJKLM)=DI(LT)*DV(IJKL1)
              CONTINUE
            DO 10004
              IJKL1=I2+L*N13
              LK1=IJKL1
              DO 10004
                IJKL1=LK1+K*N12
                KJ1=IJKL1
                DO 10004
                  IJKL1=KJ1+J*N1
                  JI1=IJKL1
                  DO 10004
                    IJKL1=JI1+I
                    IJKL3=IJKL1+N142
                    IJKL4=IJKL3+N14
                    IJKL5=IJKL4+N14
                    IJKL6=IJKL5+N14
                    V(IJKL1)=V(IJKL1)+(V(IJKL3)+2.0*(V(IJKL4)+V(IJKL5))+V(IJKL6))*
                  *F16TH
            DO 10004
              IJKL1=I2+L*N13
              LK1=IJKL1
              DO 10004
                IJKL1=LK1+K*N12
                KJ1=IJKL1
                DO 10004
                  IJKL1=KJ1+J*N1
                  JI1=IJKL1
                  DO 10004
                    IJKL1=JI1+I
                    IJKL3=IJKL1+N142
                    IJKL4=IJKL3+N14
                    IJKL5=IJKL4+N14
                    IJKL6=IJKL5+N14
                    V(IJKL1)=V(IJKL1)+(V(IJKL3)+2.0*(V(IJKL4)+V(IJKL5))+V(IJKL6))*
                  *F16TH
          CONTINUE
        RETURN
      END
10004
  
```

```

FUNCTION U(V)
INTEGER CK,CJK2
DIMENSION V(1)
COMMON/MEDIUM/VMD(400),VMDV(400),BMD(400),BMDI(400),FX(400),
*FY(400),FZ(400),HX(400),HY(400),HZ(400),ZSIDE(400),YSIDE(400),
*ZFRIT(100),ZBACK(100),YBACK(100),ZTOP(100),YTOP(100),
COMMON/SIZE/N1M1,N1,N1P1,N2M1,N2,N2P1,N3M1,N3,N3P1,N12,N13,N14,N15U
*,N23,I1,I2,I3,I4,I5,I6,I7,I8,I9,I10,I11,I12,I13,I14,I15,I16,I17,LTU
*,N132,N133,N142,N143
COMMON/TDT/T,DT(20),TS,DX,DY,DZ
C
C V(IJKLM)=V(I+(J-1)*N1+(K-1)*N12+(L-1)*N13+(M-1)*N14)
C
EN=0.
DO 1
  IJK1=I1+K*N12
  KJ5=IJK1
  IK=-N1+K*N1
  KI1=IK
  DO 1
    IJK1=KJ5+J*N1
    JI5=IJK1
    DO 1
      IJK1=JI5+I
      IJK2=IJK1+N13
      IJK3=IJK2+N13
      IJK4=IJK3+N13
      IK=KI1+I
      EN=EN+VMDV(IK)*(V(IJK2)**2+V(IJK3)**2+V(IJK4)**2)
      EN=EN+BMDI(IK)*V(IJK1)**2
    CONTINUE
  1 CONTINUE
DO 2
  IJ=-N1+J*N1
  JI=IJ
  DO 2
    IJ=JI+I
    IJ14=IJ+N133
    EN=EN+VMDV(IJ)*V(IJ14)**2
  2 CONTINUE
DO 3
  IK=-N1+K*N1
  KI=IK
  IK3=I7+K*N12
  KI1=IK3
  DO 3
    IK=KI+I
    IK3=KI1+I
    EN=EN+VMDV(IK)*V(IK3)**2
  3 CONTINUE

```

```

3  CONTINUE
   DO 4
      OK=1-N1+K*N1
      OJK2=I4+K*N12
      KJ=OJK2
   DO 4
      OJK2=KJ+J*N1
      EN=EN+VMDV(OK)*V(OJK2)**2
4  CONTINUE
   U=EN
   RETURN
   END

```

 $K=2, N3M1$ $J=2, N2MI$

```

SUBROUTINE TCP(V,C,FUN)
DIMENSION V(1),C( 8,600)
COMMON/SIZE/N1M1,N1,N1P1,N2M1,N2,N2P1,N3M1,N3,N3P1,N12,N13,N14,N15,N16,N17,L17,L18,L19,L20,L21,L22,L23,L24,L25,L26,L27,L28,L29,L30,L31,L32,L33,L34,L35,L36,L37,L38,L39,L40,L41,L42,L43,L44,L45,L46,L47,L48,L49,L50,L51,L52,L53,L54,L55,L56,L57,L58,L59,L60,L61,L62,L63,L64,L65,L66,L67,L68,L69,L70,L71,L72,L73,L74,L75,L76,L77,L78,L79,L80,L81,L82,L83,L84,L85,L86,L87,L88,L89,L90,L91,L92,L93,L94,L95,L96,L97,L98,L99,L100,L101,L102,L103,L104,L105,L106,L107,L108,L109,L110,L111,L112,L113,L114,L115,L116,L117,L118,L119,L120,L121,L122,L123,L124,L125,L126,L127,L128,L129,L130,L131,L132,L133,L134,L135,L136,L137,L138,L139,L140,L141,L142,L143,L144,L145,L146,L147,L148,L149,L150,L151,L152,L153,L154,L155,L156,L157,L158,L159,L160,L161,L162,L163,L164,L165,L166,L167,L168,L169,L170,L171,L172,L173,L174,L175,L176,L177,L178,L179,L180,L181,L182,L183,L184,L185,L186,L187,L188,L189,L190,L191,L192,L193,L194,L195,L196,L197,L198,L199,L200,L201,L202,L203,L204,L205,L206,L207,L208,L209,L210,L211,L212,L213,L214,L215,L216,L217,L218,L219,L220,L221,L222,L223,L224,L225,L226,L227,L228,L229,L230,L231,L232,L233,L234,L235,L236,L237,L238,L239,L240,L241,L242,L243,L244,L245,L246,L247,L248,L249,L250,L251,L252,L253,L254,L255,L256,L257,L258,L259,L260,L261,L262,L263,L264,L265,L266,L267,L268,L269,L270,L271,L272,L273,L274,L275,L276,L277,L278,L279,L280,L281,L282,L283,L284,L285,L286,L287,L288,L289,L290,L291,L292,L293,L294,L295,L296,L297,L298,L299,L300,L301,L302,L303,L304,L305,L306,L307,L308,L309,L310,L311,L312,L313,L314,L315,L316,L317,L318,L319,L320,L321,L322,L323,L324,L325,L326,L327,L328,L329,L330,L331,L332,L333,L334,L335,L336,L337,L338,L339,L340,L341,L342,L343,L344,L345,L346,L347,L348,L349,L350,L351,L352,L353,L354,L355,L356,L357,L358,L359,L360,L361,L362,L363,L364,L365,L366,L367,L368,L369,L370,L371,L372,L373,L374,L375,L376,L377,L378,L379,L380,L381,L382,L383,L384,L385,L386,L387,L388,L389,L390,L391,L392,L393,L394,L395,L396,L397,L398,L399,L400,L401,L402,L403,L404,L405,L406,L407,L408,L409,L410,L411,L412,L413,L414,L415,L416,L417,L418,L419,L420,L421,L422,L423,L424,L425,L426,L427,L428,L429,L430,L431,L432,L433,L434,L435,L436,L437,L438,L439,L440,L441,L442,L443,L444,L445,L446,L447,L448,L449,L450,L451,L452,L453,L454,L455,L456,L457,L458,L459,L460,L461,L462,L463,L464,L465,L466,L467,L468,L469,L470,L471,L472,L473,L474,L475,L476,L477,L478,L479,L480,L481,L482,L483,L484,L485,L486,L487,L488,L489,L490,L491,L492,L493,L494,L495,L496,L497,L498,L499,L500,L501,L502,L503,L504,L505,L506,L507,L508,L509,L510,L511,L512,L513,L514,L515,L516,L517,L518,L519,L520,L521,L522,L523,L524,L525,L526,L527,L528,L529,L530,L531,L532,L533,L534,L535,L536,L537,L538,L539,L540,L541,L542,L543,L544,L545,L546,L547,L548,L549,L550,L551,L552,L553,L554,L555,L556,L557,L558,L559,L560,L561,L562,L563,L564,L565,L566,L567,L568,L569,L570,L571,L572,L573,L574,L575,L576,L577,L578,L579,L580,L581,L582,L583,L584,L585,L586,L587,L588,L589,L590,L591,L592,L593,L594,L595,L596,L597,L598,L599,L600,L601,L602,L603,L604,L605,L606,L607,L608,L609,L610,L611,L612,L613,L614,L615,L616,L617,L618,L619,L620,L621,L622,L623,L624,L625,L626,L627,L628,L629,L630,L631,L632,L633,L634,L635,L636,L637,L638,L639,L640,L641,L642,L643,L644,L645,L646,L647,L648,L649,L650,L651,L652,L653,L654,L655,L656,L657,L658,L659,L660,L661,L662,L663,L664,L665,L666,L667,L668,L669,L670,L671,L672,L673,L674,L675,L676,L677,L678,L679,L680,L681,L682,L683,L684,L685,L686,L687,L688,L689,L690,L691,L692,L693,L694,L695,L696,L697,L698,L699,L700,L701,L702,L703,L704,L705,L706,L707,L708,L709,L710,L711,L712,L713,L714,L715,L716,L717,L718,L719,L720,L721,L722,L723,L724,L725,L726,L727,L728,L729,L730,L731,L732,L733,L734,L735,L736,L737,L738,L739,L740,L741,L742,L743,L744,L745,L746,L747,L748,L749,L750,L751,L752,L753,L754,L755,L756,L757,L758,L759,L760,L761,L762,L763,L764,L765,L766,L767,L768,L769,L770,L771,L772,L773,L774,L775,L776,L777,L778,L779,L780,L781,L782,L783,L784,L785,L786,L787,L788,L789,L790,L791,L792,L793,L794,L795,L796,L797,L798,L799,L800,L801,L802,L803,L804,L805,L806,L807,L808,L809,L810,L811,L812,L813,L814,L815,L816,L817,L818,L819,L820,L821,L822,L823,L824,L825,L826,L827,L828,L829,L830,L831,L832,L833,L834,L835,L836,L837,L838,L839,L840,L841,L842,L843,L844,L845,L846,L847,L848,L849,L850,L851,L852,L853,L854,L855,L856,L857,L858,L859,L860,L861,L862,L863,L864,L865,L866,L867,L868,L869,L870,L871,L872,L873,L874,L875,L876,L877,L878,L879,L880,L881,L882,L883,L884,L885,L886,L887,L888,L889,L890,L891,L892,L893,L894,L895,L896,L897,L898,L899,L900,L901,L902,L903,L904,L905,L906,L907,L908,L909,L910,L911,L912,L913,L914,L915,L916,L917,L918,L919,L920,L921,L922,L923,L924,L925,L926,L927,L928,L929,L930,L931,L932,L933,L934,L935,L936,L937,L938,L939,L940,L941,L942,L943,L944,L945,L946,L947,L948,L949,L950,L951,L952,L953,L954,L955,L956,L957,L958,L959,L960,L961,L962,L963,L964,L965,L966,L967,L968,L969,L970,L971,L972,L973,L974,L975,L976,L977,L978,L979,L980,L981,L982,L983,L984,L985,L986,L987,L988,L989,L990,L991,L992,L993,L994,L995,L996,L997,L998,L999,L1000,L1001,L1002,L1003,L1004,L1005,L1006,L1007,L1008,L1009,L1010,L1011,L1012,L1013,L1014,L1015,L1016,L1017,L1018,L1019,L1020,L1021,L1022,L1023,L1024,L1025,L1026,L1027,L1028,L1029,L1030,L1031,L1032,L1
```

END

TOP 0320

```

SUBROUTINE SCURCE(V,AMP,FUN)
DIMENSION V(1),AMP(8)
C THIS SUBROUTINE PRODUCES THE SIGNAL VS. TIME AT LOCATION NX5,NYS,NZ5.
C THE FREQUENCIES OF THE SIGNAL ARE FUN,2*FUN,3*FUN,...,NFREQ
COMMON/TDT/T,DT(20),TS,DX,DY,DZ
COMMON/SIG/NXYZS,NFREQ
COMMON/SIZE/N1M1,N1,N1P1,N2M1,N2,N2P1,N3M1,N3,N3P1,N12,N13,N14,N15,N17,LT
*,N23,I1,I2,I3,I4,I5,I6,I7,I8,I9,I10,I11,I12,I13,I14,I15,I16,I17,LT
*,N132,N133,N142,N143
PI2T=6.2831853*PI
V(NXYZS)=0.0
DO 1 I=1,NFREQ
  FI=I
  1 V(NXYZS)=V(NXYZS)+AMP(I)*COS(PI2T*FI*FUN)
RETURN
END
SOUR0000
SOUR0010
SOUR0020
SOUR0030
SOUR0040
SOUR0050
SOUR0060
SOUR0070
SOUR0080
SOUR0090
SOUR0100
SOUR0110
SOUR0120
SOUR0130
SOUR0140
SOUR0150

```

```

SUBROUTINE BACK(V,B,FUN)
DIMENSION V(1),B( 8,600)
INTEGER CPJKL
COMMON/SIZE/N1M1,N1,N1P1,N2M1,N2,N2P1,N3M1,N3,N3P1,N12,N13,N14,N15,N17,LT
*,N23,I1,I2,I3,I4,I5,I6,I7,I8,I9,I10,I11,I12,I13,I14,I15,I16,I17,LT
*,N132,N133,N142,N143
COMMON/TDT/T,DT(20),TS,DX,DY,DZ
COMMON/SIG/NXYZS,NFREQ
C V(IJKLM)=V(I+(J-1)*N1+(K-1)*N12+(L-1)*N13+(M-1)*N14)
C
C
PI2T=6.2831853*PI
DO 1 L=1,4
  OPJKL=I9+L*N13
  LK=OPJKL
  JKLE=I6+L*N23
  LK1=JKLE
  DO 1 K=2,N3M1
    OPJKL=LK+K*N12
    KJ=OPJKL
    JKLE=LK1+K*N2
    KJ1=JKLE
  DO 1 J=2,N2M1
    BACK0000
    BACK0010
    BACK0020
    BACK0030
    BACK0040
    BACK0050
    BACK0060
    BACK0070
    BACK0080
    BACK0090
    BACK0100
    BACK0110
    BACK0120
    BACK0130
    BACK0140
    BACK0150
    BACK0160
    BACK0170
    BACK0180
    BACK0190
    BACK0200
    BACK0210
    BACK0220

```


BACK0230
 BACK0240
 BACK0250
 BACK0260
 BACK0270
 BACK0280
 BACK0290
 BACK0300
 BACK0310
 BACK0320
 HARM0000
 HARM0010
 HARM0020
 HARM0030
 HARM0040
 HARM0050
 HARM0060
 HARM0070
 HARM0080
 HARM0090
 HARM0100
 HARM0110
 HARM0120
 HARM0130
 HARM0140
 HARM0150
 HARM0160
 HARM0170
 HARM0180
 HARM0190
 HARM0200
 HARM0210
 HARM0220
 HARM0230
 HARM0240
 HARM0250
 HARM0260
 HARM0270
 HARM0280
 HARM0290
 HARM0300
 HARM0310
 HARM0320
 HARM0330
 HARM0340
 HARM0350
 HARM0360
 HARM0370

```

OPJKL=KJ+J*N1
JKLE=2*(KJ1+J)
JKLO=JKLE-1
V(OPJKL)=0.0
DO 1 I=2,NFREQ
  FIN=IN-1
  V(OPJKL)=V(OPJKL)+B(IN,JKLC)*COS(PI2T*FIN*FUN +B(IN,JKLE))
  RETURN
END
1

```

```

.....
SURCLUTINE HARM
PURPOSE
PERFORMS DISCRETE COMPLEX FOURIER TRANSFORMS ON A COMPLEX
THREE DIMENSIONAL ARRAY
CALL HARM (A,M,INV,S,IFSET,IFERR)
USAGE
DESCRIPTION OF PARAMETERS
A - AS INPUT, A CONTAINS THE COMPLEX, 3-DIMENSIONAL
  ARRAY TO BE TRANSFORMED. THE REAL PART OF
  A(I1,I2,I3) IS STCRED IN VECTOR FASHION IN A CELL
  WITH INDEX 2*(I3*N1*N2 + I2*N1 + I1) + 1 WHERE
  NI = 2**M(I), I=1,2,3 AND I1=0,1,...,NI-1 ETC.
  THE IMAGINARY PART IS IN THE CELL IMMEDIATELY
  FOLLOWING. NOTE THAT INCREASES IN I1 INCREASES
  MOST RAPIDLY AND I3 INCREASES LEAST RAPIDLY.
  AS OUTPUT, A CONTAINS THE COMPLEX FOURIER
  TRANSFORM. A CONTAINS THE CORE LOCATIONS OF
  ARRAY A IS 2*(N1*N2*N3)
M - A THREE CELL VECTOR WHICH DETERMINES THE SIZES
  OF THE 3 DIMENSIONS OF THE ARRAY A. THE SIZE,
  NI, OF THE I DIMENSION OF A IS 2**M(I), I = 1,2,3
INV - A VECTOR WORK AREA FOR BIT AND INDEX MANIPULATION
  OF DIMENSION ONE EIGHTH THE NUMBER OF CORE
  LOCATIONS OF A, VIZ., (1/8)*2*N1*N2*N3
S - A VECTOR WORK AREA FOR SINE TABLES WITH DIMENSION
  THE SAME AS INV
IFSET - AN OPTICN PARAMETER WITH THE FOLLOWING SETTINGS
  0 SET UP SINE AND INV TABLES ONLY
  1 SET UP SINE AND INV TABLES ONLY AND
    CALCULATE FOURIER TRANSFORM
    -1 CALCULATE SINE AND INV TABLES ONLY AND
      CALCULATE INVERSE FOURIER TRANSFORM (FOR
        THE MEANING OF INVERSE SEE THE EQUATIONS

```

CC

HARM0380
HARM0390
HARM0400
HARM0410
HARM0420
HARM0430
HARM0440
HARM0450
HARM0460
HARM0470
HARM0480
HARM0490
HARM0500
HARM0510
HARM0520
HARM0530
HARM0540
HARM0550
HARM0560
HARM0570
HARM0580
HARM0590
HARM0600
HARM0610
HARM0620
HARM0630
HARM0640
HARM0650
HARM0660
HARM0670
HARM0680
HARM0690
HARM0700
HARM0710
HARM0720
HARM0730
HARM0740
HARM0750
HARM0760
HARM0770
HARM0780
HARM0790
HARM0800
HARM0810
HARM0820
HARM0830
HARM0840
HARM0850

```

      2      UNDER METHOD BELOW)
      -2      CALCULATE FOURIER TRANSFORM ONLY (ASSUME
              SINE AND INV TABLES EXIST)
              CALCULATE INVERSE FOURIER TRANSFORM ONLY
              (ASSUME SINE AND INV TABLES EXIST)
              IFERR = 1 MEANS THE MAXIMUM M(I) IS LESS THAN 3
              OR GREATER THAN 20, I=1,2,3 WHEN IFSET IS
              +2,-2, IFERR = 1 MEANS THAT THE SINE AND INV
              TABLES ARE NOT LARGE ENOUGH OR HAVE NOT BEEN
              COMPUTED. IF ON RETURN IFERR = 0 THEN NONE OF
              THE ABOVE CONDITIONS ARE PRESENT
  
```

REMARKS
THIS SUBROUTINE IS TO BE USED FOR COMPLEX, 3-DIMENSIONAL
ARRAYS IN WHICH EACH DIMENSION IS A POWER OF 2. THE
MAXIMUM M(I) MUST NOT BE LESS THAN 3 OR GREATER THAN 20,
I = 1,2,3

SUBROUTINES AND FUNCTION SUBPROGRAMS REQUIRED
NCNE

METHOD
FOR IFSET = +1, OR +2, THE FOURIER TRANSFORM OF COMPLEX
ARRAY A IS OBTAINED.

```

      X(J1,J2,J3)=SUM      N1-1      N2-1      N3-1      L1      L2      L3
      K1=0      SUM      K2=0      K3=0      A(K1,K2,K3)*W1 *W2 *W3
  
```

WHERE W1 IS THE N(I) ROOT OF UNITY AND L1=K1*J1,
L2=K2*J2, L3=K3*J3

FOR IFSET = -1, OR -2, THE INVERSE FOURIER TRANSFORM A OF
COMPLEX ARRAY X IS OBTAINED.

```

      A(K1,K2,K3)=
      1      N1-1      N2-1      N3-1      -L1      -L2      -L3
      ----- *SUM      SUM      SUM      X(K1,K2,K3)*W1 *W2 *W3
      N1*N2*N3      J1=0      J2=0      J3=0
  
```

SEE J.W. COOLEY AND J.W. TUKEY, "AN ALGORITHM FOR THE
MACHINE CALCULATION OF COMPLEX FOURIER SERIES",
MATHEMATICS OF COMPUTATIONS, VOL. 19 (APR. 1965), P. 297.


```

SUBROUTINE HARM(A,M,INV,S,IFSET,IFERR)
DIMENSION A(1),INV(1),S(1),N(3),M(3),NP(3),W(2),W2(2),W3(2)
EQUIVALENCE (N1,N(1)),(N2,N(2)),(N3,N(3))
IF( IABS(IFSET) - 1) 900,900,12
MTT=MAXO(M(1),M(2),M(3)) -2
ROOT2 = SORT(2.)
IF (MTT-MT) 14,14,13
13 IFERR=1
14 RETURN
14 IFERR=0
M1=M(1)
M2=M(2)
M3=M(3)
N1=2**M1
N2=2**M2
N3=2**M3
16 IF(IFSET) 18,18,20
18 NX= N1*N2*N3
FN = NX
DO 19 I = 1,NX
A(2*I-1) = A(2*I-1)/FN
19 A(2*I) = -A(2*I)/FN
20 NP(1)=N1*N2
NP(2)=NP(1)*N3
DO 250 ID=1,3
IL = NP(3)-NP(ID)
IL1 = IL+1
MI = M(ID)
IF (MI)250,250,3C
30 IDIF=NP(ID)
KBIT=NP(ID)
MEV = 2*(MI/2)
IF (MI - MEV)60,60,40
C
40 M IS ODD. DO L=1 CASE
KBIT=KBIT/2
KL=KBIT-2
DO 50 I=1,IL1,IDIF
KLAST=KL+I
DO 50 K=1,KLAST,2
KD=K+KBIT
C
DO ONE STEP WITH L=1,J=0

```

HARM0860
HARM0870
HARM0880
HARM0890
HARM0900
HARM0910
HARM0920
HARM0930
HARM0940
HARM0950
HARM0960
HARM0970
HARM0980
HARM0990
HARM1000
HARM1010
HARM1020
HARM1030
HARM1040
HARM1050
HARM1060
HARM1070
HARM1080
HARM1090
HARM1100
HARM1110
HARM1120
HARM1130
HARM1140
HARM1150
HARM1160
HARM1170
HARM1180
HARM1190
HARM1200
HARM1210
HARM1220
HARM1230
HARM1240
HARM1250
HARM1260
HARM1270
HARM1280
HARM1290

HARM1300
HARM1310
HARM1320
HARM1330
HARM1340
HARM1350
HARM1360
HARM1370
HARM1380
HARM1390
HARM1400
HARM1410
HARM1420
HARM1430
HARM1440
HARM1450
HARM1460
HARM1470
HARM1480
HARM1490
HARM1500
HARM1510
HARM1520
HARM1530
HARM1540
HARM1550
HARM1560
HARM1570
HARM1580
HARM1590
HARM1600
HARM1610
HARM1620
HARM1630
HARM1640
HARM1650
HARM1660
HARM1670
HARM1680
HARM1690
HARM1700
HARM1710
HARM1720
HARM1730
HARM1740
HARM1750
HARM1760
HARM1770

```

C      A(K)=A(K)+A(KD)
C      A(KD)=A(K)-A(KD)
C      T=A(KD)
      A(KD)=A(K)-T
      A(K)=A(K)+T
      T=A(KD+1)
      A(KD+1)=A(K+1)-T
      A(K+1)=A(K+1)+T
50    IF (MI - 1)250,250,52
52    LFIRST = 3
C      DEF - JLAST = 2** (L-2) -1
C      JLAST=1
C      GO TO 7C
C      M IS EVEN
60    LFIRST = 2
C      JLAST=0
70    DO 240 L=LFIRST,MI,2
      JJDIFF=KBIT
      KBIT=KBIT/4
      KL=KBIT-2
C      DO FOR J=0
C      DO 80 I=1,IL1,IDIF
C      KLAST=I+KL
C      DO 80 K=I,KLAST,2
C      K1=K+KBIT
C      K2=K1+KBIT
C      K3=K2+KBIT
C      DO TWO STEPS WITH J=0
C      A(K)=A(K)+A(K2)
C      A(K2)=A(K)-A(K2)
C      A(K1)=A(K1)+A(K3)
C      A(K3)=A(K1)-A(K3)
C      A(K)=A(K)+A(K1)
C      A(K1)=A(K)-A(K1)
C      A(K2)=A(K2)+A(K3)*I
C      A(K3)=A(K2)-A(K3)*I
C      T=A(K2)
C      A(K2)=A(K)-T
C      A(K)=A(K)+T
C      T=A(K2+1)
C      A(K2+1)=A(K+1)-T

```

| | | |
|----|---|----------|
| C | A(K+1)=A(K+1)+T | HARM1780 |
| | T=A(K3) | HARM1790 |
| | A(K3)=A(K1)-T | HARM1800 |
| | A(K1)=A(K1)+T | HARM1810 |
| | T=A(K3+1) | HARM1820 |
| | A(K3+1)=A(K1+1)-T | HARM1830 |
| | A(K1+1)=A(K1+1)+T | HARM1840 |
| C | T=A(K1) | HARM1850 |
| | A(K1)=A(K)-T | HARM1860 |
| | A(K)=A(K)+T | HARM1870 |
| | T=A(K1+1) | HARM1880 |
| | A(K1+1)=A(K+1)-T | HARM1890 |
| | A(K+1)=A(K+1)+T | HARM1900 |
| C | R=-A(K3+1) | HARM1910 |
| | T=A(K3) | HARM1920 |
| | A(K3)=A(K2)-R | HARM1930 |
| | A(K2)=A(K2)+R | HARM1940 |
| | A(K3+1)=A(K2+1)-T | HARM1950 |
| | A(K2+1)=A(K2+1)+T | HARM1960 |
| 80 | IF (JLAST) 235,235,82 | HARM1970 |
| 82 | JJ=JJ+1 | HARM1980 |
| C | DO FOR J=1 | HARM1990 |
| | ILAST=IL+JJ | HARM2000 |
| C | DO 85 I=JJ,ILAST,1DIF | HARM2010 |
| | KLAST=KL+I | HARM2020 |
| | DO 85 K=I,KLAST,2 | HARM2030 |
| | K1=K+KBIT | HARM2040 |
| | K2=K1+KBIT | HARM2050 |
| | K3=K2+KBIT | HARM2060 |
| C | LETting W=(1+I)/ROOT2,W3=(-1+I)/ROOT2,W2=I, | HARM2070 |
| | A(K)=A(K)+A(K2)*I | HARM2080 |
| | A(K2)=A(K)-A(K2)*I | HARM2090 |
| | A(K1)=A(K1)*W+A(K3)*W3 | HARM2100 |
| | A(K3)=A(K1)*W-A(K3)*W3 | HARM2110 |
| C | A(K)=A(K)+A(K1) | HARM2120 |
| | A(K1)=A(K)-A(K1) | HARM2130 |
| | A(K2)=A(K2)+A(K3)*I | HARM2140 |
| | A(K3)=A(K2)-A(K3)*I | HARM2150 |
| C | R=-A(K2+1) | HARM2160 |
| | T=A(K2) | HARM2170 |
| | A(K2)=A(K)-R | HARM2180 |
| | | HARM2190 |
| | | HARM2200 |
| | | HARM2210 |
| | | HARM2220 |
| | | HARM2230 |
| | | HARM2240 |
| | | HARM2250 |

HARM 2260
HARM 2270
HARM 2280
HARM 2290
HARM 2300
HARM 2310
HARM 2320
HARM 2330
HARM 2340
HARM 2350
HARM 2360
HARM 2370
HARM 2380
HARM 2390
HARM 2400
HARM 2410
HARM 2420
HARM 2430
HARM 2440
HARM 2450
HARM 2460
HARM 2470
HARM 2480
HARM 2490
HARM 2500
HARM 2510
HARM 2520
HARM 2530
HARM 2540
HARM 2550
HARM 2560
HARM 2570
HARM 2580
HARM 2590
HARM 2600
HARM 2610
HARM 2620
HARM 2630
HARM 2640
HARM 2650
HARM 2660
HARM 2670
HARM 2680
HARM 2690
HARM 2700
HARM 2710
HARM 2720
HARM 2730

```

A(K) = A(K)+R
A(K2+1)=A(K+1)-T
A(K+1)=A(K+1)+T
AWR=A(K1)-A(K1+1)
AWI = A(K1+1)+A(K1)
R=-A(K3)-A(K3+1)
T=A(K3)-A(K3+1)
A(K3)=(AWR-R)/ROOT2
A(K3+1)=(AWI-T)/ROOT2
A(K1)=(AWR+R)/ROOT2
A(K1+1)=(AWI+T)/ROOT2
T=A(K1)
A(K1)=A(K)-T
A(K)=A(K)+T
T=A(K1+1)
A(K1+1)=A(K+1)-T
A(K+1)=A(K+1)+T
R=-A(K3+1)
T=A(K3)
A(K3)=A(K2)-R
A(K2)=A(K2)+R
A(K3+1)=A(K2+1)-T
A(K2+1)=A(K2+1)+T
85 IF(JLAST-1) 235,235,90
90 JJ= JJ + JJDIFF
NOW DO THE REMAINING J'S
DO 230 J=2,JLAST
C
C
C
C
FETCH W'S
DEF- W=W*INV(J), W2=W**2, W3=W**3
96 I=INV(J+1)
98 IC=NT-I
W(1)=S(IC)
W(2)=S(I)
I2=2*I
I2C=NT-I2
IF(I2C)I2C,I10,I0C
C
C
2*I IS IN FIRST QUADRANT
100 W2(1)=S(I2C)
W2(2)=S(I2)
GO TO 130
110 W2(1)=C.
W2(2)=1.
GO TO 130
C

```

```

C      120      2*I IS IN SECOND QUADRANT
I2CC=I2C+NT
I2C=-I2C
W2(1)=-S(I2C)
W2(2)=S(I2CC)
C      130      I3=I+I2
I3C=NT-I3
IF(I3C)160,150,140
C      140      I3 IN FIRST QUADRANT
W3(1)=S(I3C)
W3(2)=S(I3)
GO TO 200
C      150      W3(1)=0.
W3(2)=1.
GO TO 200
C      160      I3CC=I3C+NT
IF(I3CC)190,180,170
C      170      I3 IN SECOND QUADRANT
I3C=-I3C
W3(1)=-S(I3C)
W3(2)=S(I3CC)
GO TO 200
C      180      W3(1)=-1.
W3(2)=0.
GO TO 200
C      190      3*I IN THIRD QUADRANT
I3CC=NT+I3CC
I3C=-I3CC
W3(1)=-S(I3CCC)
W3(2)=-S(I3CC)
C      200      ILAST=IL+JJ
DO 220 I=JJ, ILAST, IDIF
KLAST=KL+I
DO 220 K=I, KLAST, 2
K1=K+KBIT
K2=K1+KBIT
K3=K2+KBIT
C      DO TWO STEPS WITH J NCT 0
A(K)=A(K)+A(K2)*W2
A(K2)=A(K)-A(K2)*W2
A(K1)=A(K1)*W+A(K3)*W3
A(K3)=A(K1)*W-A(K3)*W3
C

```

```

HARM2740
HARM2750
HARM2760
HARM2770
HARM2780
HARM2790
HARM2800
HARM2810
HARM2820
HARM2830
HARM2840
HARM2850
HARM2860
HARM2870
HARM2880
HARM2890
HARM2900
HARM2910
HARM2920
HARM2930
HARM2940
HARM2950
HARM2960
HARM2970
HARM2980
HARM2990
HARM3000
HARM3010
HARM3020
HARM3030
HARM3040
HARM3050
HARM3060
HARM3070
HARM3080
HARM3090
HARM3100
HARM3110
HARM3120
HARM3130
HARM3140
HARM3150
HARM3160
HARM3170
HARM3180
HARM3190
HARM3200
HARM3210

```

HARM3220
HARM3230
HARM3240
HARM3250
HARM3260
HARM3270
HARM3280
HARM3290
HARM3300
HARM3310
HARM3320
HARM3330
HARM3340
HARM3350
HARM3360
HARM3370
HARM3380
HARM3390
HARM3400
HARM3410
HARM3420
HARM3430
HARM3440
HARM3450
HARM3460
HARM3470
HARM3480
HARM3490
HARM3500
HARM3510
HARM3520
HARM3530
HARM3540
HARM3550
HARM3560
HARM3570
HARM3580
HARM3590
HARM3600
HARM3610
HARM3620
HARM3630
HARM3640
HARM3650
HARM3660
HARM3670
HARM3680
HARM3690

```

C      A(K)=A(K)+A(K1)
C      A(K1)=A(K)-A(K1)
C      A(K2)=A(K2)+A(K3)*I
C      A(K3)=A(K2)-A(K3)*I
C
C      R=A(K2)*W2(1)-A(K2+1)*W2(2)
C      T=A(K2)*W2(2)+A(K2+1)*W2(1)
C      A(K2)=A(K)-R
C      A(K)=A(K)+R
C      A(K2+1)=A(K+1)-T
C      A(K+1)=A(K+1)+T
C
C      R=A(K3)*W3(1)-A(K3+1)*W3(2)
C      T=A(K3)*W3(2)+A(K3+1)*W3(1)
C      AWR=A(K1)*W(1)-A(K1+1)*W(2)
C      AWI=A(K1)*W(2)+A(K1+1)*W(1)
C      A(K3)=AWR-R
C      A(K3+1)=AWI-T
C      A(K1)=AWR+R
C      A(K1+1)=AWI+T
C      T=A(K1)
C      A(K1)=A(K)-T
C      A(K)=A(K)+T
C      T=A(K1+1)
C      A(K1+1)=A(K+1)-T
C      A(K+1)=A(K+1)+T
C      R=-A(K3+1)
C      T=A(K3)
C      A(K3)=A(K2)-R
C      A(K2)=A(K2)+R
C      A(K3+1)=A(K2+1)-T
C      A(K2+1)=A(K2+1)+T
C      END OF I AND K LOOPS
220
C
C      JJ=JJDIF+JJ
C      END OF J-LCOP
C
C      JLAST=4*JLAST+3
C      CONTINUE
C      END OF L LCOP
C
C      CONTINUE
C      END OF ID LOOP
C
C      WE NOW HAVE THE COMPLEX FOURIER SUMS BUT THEIR ADDRESSES ARE
C      BIT-REVERSED. THE FOLLOWING ROUTINE PUTS THEM IN ORDER
C      NTSQ=NT*NT
C      M3MT=M3-MT

```



```

350 IF(M3MT) 37C,360,36C
C
360 M3 GR. CR EQ. MT
      IGO3=1
      N3VNT=N3/NT
      MINN3=NT
      GO TO 380
C
C
370 M3 LESS THAN MT
      IGO3=2
      N3VNT=1
      NTVN3=NT/N3
      MINN3=N3
      JJD3 = NTSQ/N3
      M2MT=M2-MT
450 IF (M2MT)470,460,460
C
C
460 M2 GR. OR EQ. MT
      IGO2=1
      N2VNT=N2/NT
      MINN2=NT
      GO TO 48C
C
C
470 M2 LESS THAN MT
      IGO2 = 2
      N2VNT=1
      NTVN2=NT/N2
      MINN2=N2
      JJD2=NTSQ/N2
      M1MT=M1-MT
550 IF(M1MT)570,560,560
C
C
560 M1 GR. OR EQ. MT
      IGO1=1
      N1VNT=N1/NT
      MINN1=NT
      GO TO 58C
C
C
570 M1 LESS THAN MT
      IGO1=2
      N1VNT=1
      NTVN1=NT/N1
      MINN1=N1
      JJD1=NTSQ/N1
      JJD1=1
      J=1
      DO 880 JPP3=1,N3VNT
      IPP3=INV(JJ3)
580
600

```

```

HARM3700
HARM3710
HARM3720
HARM3730
HARM3740
HARM3750
HARM3760
HARM3770
HARM3780
HARM3790
HARM3800
HARM3810
HARM3820
HARM3830
HARM3840
HARM3850
HARM3860
HARM3870
HARM3880
HARM3890
HARM3900
HARM3910
HARM3920
HARM3930
HARM3940
HARM3950
HARM3960
HARM3970
HARM3980
HARM3990
HARM4000
HARM4010
HARM4020
HARM4030
HARM4040
HARM4050
HARM4060
HARM4070
HARM4080
HARM4090
HARM4100
HARM4110
HARM4120
HARM4130
HARM4140
HARM4150
HARM4160
HARM4170

```

```

      DO 870 JP3=1,MINN3
      GO TO (61C,620),IG03
      IP3=INV(JP3)*N3 VNT
      61C GO TO 630
      62C IP3=INV(JP3)/NTVN3
      630 I3=(IPP3+IP3)*N2
      700 JJ2=1
      DO 870 JPP2=1,N2VNT
      IPP2=INV(JJ2)+I3
      DO 860 JP2=1,MINN2
      GO TO (710,720),IGC2
      710 IP2=INV(JP2)*N2 VNT
      GO TO 730
      720 IP2=INV(JP2)/NTVN2
      730 I2=(IPP2+IP2)*N1
      800 JJ1=1
      DO 860 JPP1=1,N1VNT
      IPP1=INV(JJ1)+I2
      DO 850 JP1=1,MINN1
      GO TO (81C,820),IGC1
      810 IP1=INV(JP1)*N1 VNT
      GO TO 830
      820 IP1=INV(JP1)/NTVN1
      830 I1=2*(IPP1+IP1)+1
      IF (J-I) 840,845,845
      94C T=A(I)
      A(I)=A(J)
      A(J)=T
      T=A(I+1)
      A(I+1)=A(J+1)
      A(J+1)=T
      CONTINUE
      J=J+2
      845 JJ1=JJ1+JJ01
      850 END OF JPP1 AND JP2
      860
      C C
      870 JJ2=JJ2+JJ02
      END OF JPP2 AND JP3 LCOPS
      C C
      880 JJ3 = JJ3+JJ03
      END OF JPP3 LOOP
      C C
      890 IF (IFSET) 891,895,895
      891 DO 892 I = 1,NX
      892 A(2*I) = -A(2*I)
      895 RETURN
      C C

```

THE FOLLOWING PROGRAM COMPUTES THE SIN AND INV TABLES.

HARM418C
HARM4190
HARM4200
HARM4210
HARM4220
HARM4230
HARM4240
HARM4250
HARM4260
HARM427C
HARM4280
HARM4290
HARM4300
HARM4310
HARM4320
HARM4330
HARM4340
HARM4350
HARM4360
HARM4370
HARM4380
HARM4390
HARM4400
HARM4410
HARM4420
HARM4430
HARM444C
HARM4450
HARM4460
HARM4470
HARM448C
HARM449C
HARM4500
HARM4510
HARM4520
HARM4530
HARM4540
HARM4550
HARM4560
HARM4570
HARM4580
HARM4590
HARM4600
HARM4610
HARM4620
HARM4630
HARM464C
HARM4650

```

C      900 MT=MAXO(M(1),M(2),M(3)) -2
      904 MT = MAXO(2,MT)
      905 IF (MT-2C)906,906,905
      906 IFERR = 1
      GO TO 895
      IFERR=0
      NT=2**MT
      NTV2=NT/2
C      SET UP SIN TABLE
C      THETA=PIE/2**(L+1) FOR L=1
C      910 THETA=.7853981634
C      JSTEP=2**(MT-L+1) FOR L=1
C      JSTEP=NT
C      JDIF=2**(MT-L) FOR L=1
C      JDIF=2**(MT-L) FOR L=1
C      S(JDIF)=SIN(THETA)
C      JDIF=NTV2
C      DO 950 L=2,MT
C      THETA=THETA/2.
C      JSTEP2=JSTEP
C      JSTEP=JDIF
C      JDIF=JSTEP/2
C      S(JDIF)=SIN(THETA)
C      JC1=NT-JDIF
C      S(JC1)=CCS(THETA)
C      JLAST=NT-JSTEP2
C      IF(JLAST - JSTEP) 950,920,920
C      920 DO 940 J=JSTEP,JLAST,JSTEP
C      JC=NT-J
C      JD=J+JDIF
C      940 S(JD)=S(J)*S(JC1)+S(JDIF)*S(JC)
C      950 CONTINUE
C      SET UP INV(J) TABLE
C      960 MTEXP=NTV2
C      MTEXP=2**(MT-L). FOR L=1
C      LM1EXP=1
C      LM1EXP=2**(L-1). FOR L=1
C      INV(1)=0
C      DO 980 L=1,MT
C      INV(LM1EXP+1) = MTEXP

```

HARM4660
 HARM467C
 HARM468C
 HARM4690
 HARM4700
 HARM4710
 HARM472C
 HARM4730
 HARM4740
 HARM4750
 HARM476C
 HARM4770
 HARM478C
 HARM4790
 HARM4800
 HARM4810
 HARM4820
 HARM4830
 HARM4840
 HARM4850
 HARM4860
 HARM4870
 HARM4880
 HARM489C
 HARM4900
 HARM4910
 HARM4920
 HARM4930
 HARM4940
 HARM4950
 HARM4960
 HARM4970
 HARM4980
 HARM4990
 HARM5000
 HARM5010
 HARM5020
 HARM5030
 HARM5040
 HARM5050
 HARM5060
 HARM5070
 HARM5080
 HARM5090
 HARM5100
 HARM511C
 HARM5120
 HARM5130

HARM5140
HARM5150
HARM5160
HARM5170
HARM5180
HARM5190
HARM5200

DERFU000
DERFU001

DERFU006
DERFU011
DERFU012
MAIN C00
MAIN 002
MAIN 003

RKAM 000
RKAM 001
RKAM 002
RKAM 003
RKAM 004

RKAM 007
RKAM 008
RKAM 009
RKAM 010

```

DO 970 J=2, LM1EXP
  JJ=J+LM1EXP
  INV(JJ)=INV(J)+MTLEXP
970  MTLEXP=MTLEXP/2
980  LM1EXP=LM1EXP*2
982  IF(IFSET)12,895,12
      END

```

```

SUBROUTINE DERFUN(Y,DY,T)
  DIMENSION Y(1),DY(1)
  DATA C/100./
  DY(1)=0.
  DO 5 I=2,200
    DY(I)=-C*(Y(I+1)-Y(I-1))
5  CONTINUE
  DY(201)=DY(200)
  RETURN
END
DOUBLE PRECISION UY
DIMENSION UY(1005),VF(1005),YD(808),Y(201),DY(201)
CALL RKAM(Y,DY,YD,VF,UY)
END

```

```

SUBROUTINE RKAM(Y,DY,YD,VF,UY)
  DOUBLE PRECISION UY
  DIMENSION Y(1),UY(1),VF(1),YD(1),A(20),D(20),ID(20),B(20),DY(1)
  *,XV(5),BET(4),IOR(10),IOW(10),JXY(10),YX(101,6),MY(6),YY(101,6),
  *,IYX(4),IYY(4),IYG(101),YVG(101),LAB(1)
  DIMENSION XX(101),YYG(101),LAB(1)
  EQUIVALENCE (M1,NN), (M2,MODE), (A2,ERMAX), (A3,ERMIN), (A4,STMAX),
  *(A5,STMIN), (A6,FACT)
  EQUIVALENCE (XX(1),YX(1,1)), (YVG(1),YX(1,2))
  REAL IT(24)/56HP(X,T) VS. X FOR T= SEC. DX=0.01 R./
  *JOHNSON BOX 70
  DEFINE FILE 8(5,100,E,J9)
  DATA LA/4H
  DATA IOR/4H(8E1,4H0.3),8*4H /,IOW/4H(1H+,4H,11X,4H6(14,4H,1PE,4
  *H12.4,4H)/(1,4H2X,6,4H(14,4HE12.,4H4))//
  DATA BET/0.5,0.5,1.0,C.0/,MY/1,2,3,4,5,6/,JXY/1,2,1,3,1,4,1,5,1,
  *6/,IY/3H Y(/,MP/3H Y(/
  C RUNGE-KUTTA ADAMS MOULTON SOLUTION OF N SIMULTANEOUS FIRST ORDER
  C ORDINARY DIFFERENTIAL EQUATIONS USING PARTIAL DOUBLE PRECISION ARITH-

```


RKAM 053
RKAM 056
RKAM 057
RKAM 058
RKAM 051

WRITE(6,90)
WRITE(6,120)A1,A2,A3,A4,A5,A6
WRITE(6,60)ALPHA,OMEGA
WRITE(6,110)
WRITE(6,IOW)(I,Y(I),I=1,M1)
WRITE(6,40)(IOW(I),I=1,IOW(J),J=1,IC)

C GRAPH READ STATEMENTS*****
C

331 IF(IYVSX)335,335,331
331 READ(5,20)NYX,(IYX(I),I=1,NYX)
335 WRITE(6,50)NYX,(IYP,IYX(I),I=1,NYX)
336 IF(IYVSY)337,337,336
336 READ(5,20)NYV,(IYV(2*I-1),IYV(2*I),I=1,NYV)
337 WRITE(6,150)(MP,IYV(2*I-1),IYV(2*I),I=1,NYV)
337 CONTINUE
LL=1
KM=1
INP=0
IND=0
JNP=0
JND=C
T=ALPHA
WRITE(6,220)

C59
RKAM 060
RKAM 061
RKAM 062
RKAM 063
RKAM 064
RKAM 065
RKAM 066
RKAM 067
RKAM 068
RKAM 069
RKAM 070
RKAM 071
RKAM 072
RKAM 073
RKAM 074
RKAM 075
RKAM 076
RKAM 077
RKAM 078
RKAM 079
RKAM 080
RKAM 081
RKAM 082
RKAM 083
RKAM 084
RKAM 085
RKAM 086
RKAM 087
RKAM 088
RKAM 089
RKAM 090
RKAM 091

C SET CONSTANTS ON RUNGE-KUTTA AND (IF USED), ADAMS-MCULTON*****
C

7 F16TH=1./6.
F24=1./24.
GO TO (7,88,88),M2
MM=4
J1=4
GO TO 9
88 MM=1
J1=1
9 EPM=2.0*ABS(CMEGA-ALPHA)
E3=EPM
ZOT=.5+2.0*(-30.0)
N2=NN+2
DT=A1
R=19.0/270.0
XV(MM)=T
IF(ERMIN)11,11,13
11 ERMIN=ERMAX/55.
13 IF(FACT)14,14,15
14 FACT=0.5
15 CALL DERFUN(Y,OY,T)
MMM=(MM-1)*NN

RKAM 097


```

DO 320 I=1,NN
IMM=MMM+I
VF(IMM)=DY(I)
UY(IMM)=Y(I)
320 GO TO (501,502),LT
501 E3=ABS(A(I))-B(I)
504 IF(D(I))505,504,505
505 E3=ABS(DI*0.5)
ABM=E3
WRITE(6,400)KM,A(KM),D(KM),B(KM)
WRITE(6,130)
GO TO 1001
502 WRITE(6,100)

C RUNGE-KUTTA ROUTINE*****
C
1001 MMM=(MM-1)*NN
KI=NN
DO 1034 K=1,4
KI=KI+NN
DO 1350 I=1,NN
IK=KI+I
IMM=MMM+I
YD(IK)=DI*VF(IMM)
1350 Y(I)=UY(IMM)+BET(K)*YD(IK)
T=BET(K)*DI+XV(MM)
CALL DERFUN(Y,DY,T)
DO 1100 I=1,NN
VF(I+MMM)=DY(I)
1100 CONTINUE
DO 1039 I=1,NN
I2=I+NN
I3=I2+NN
I4=I3+NN
IMM=MMM+I
UY(IMM+NN)=UY(IMM)+(YD(I)+2.*(YD(I2)+YD(I3))+YD(I4))*F16TH
1039 CONTINUE
MM=MM+1
XV(MM)=XV(MM-1)+DI
MMM=(MM-1)*NN
DO 1400 I=1,NN
Y(I)=UY(MM+I)
1400 T=XV(MM)
CALL DERFUN(Y,DY,T)
GO TO (42,99,99),MODE
99 MMM=(MM-1)*NN
DO 151 I=1,NN
151 VF(MM+I)=DY(I)

```

RKAM 098
 RKAM 099
 RKAM 100
 RKAM 101
 RKAM 102
 RKAM 103
 RKAM 104
 RKAM 105
 RKAM 106
 RKAM 107
 RKAM 108
 RKAM 109
 RKAM 110
 RKAM 111
 RKAM 112
 RKAM 113
 RKAM 114
 RKAM 115
 RKAM 116
 RKAM 117
 RKAM 118
 RKAM 119
 RKAM 120
 RKAM 121
 RKAM 122
 RKAM 123
 RKAM 124
 RKAM 125
 RKAM 126
 RKAM 127
 RKAM 128
 RKAM 129
 RKAM 130
 RKAM 131
 RKAM 132
 RKAM 133
 RKAM 134
 RKAM 135
 RKAM 136
 RKAM 137
 RKAM 138
 RKAM 139
 RKAM 140
 RKAM 141
 RKAM 142
 RKAM 143
 RKAM 144
 RKAM 145

```

      IF(MM-3)1001,1001,2000
C ADAMS-MOULTCN ROUTINE*****
C
2000 DO 2048 I=1,NN
      I2=I+NN
      I3=I2+NN
      I4=I3+NN
      I5=I4+NN
      Y(I)=UY(I4)+DT*(55.*VF(I4)-59.*VF(I3)+37.*VF(I2)-9.*VF(I))*F24
2048 YD(I)=Y(I)
      T=XV(4)+DT
      CALL DERFUN(Y,DY,T)
      XV(5)=T
      DO 2051 I=1,NN
      I2=I+NN
      I3=I2+NN
      I4=I3+NN
      I5=I4+NN
      UY(I5)=UY(I4)+DT*(9.*DY(I)+19.*VF(I4)-5.*VF(I3)+VF(I2))*F24
2051 Y(I)=UY(I5)
      CALL DERFUN(Y,DY,T)
      GO TO (42,42,3000),MODE
C ERROR CHECKING ROUTINE*****
C
3000 SSE=C.0
      DO 3033 I=1,NN
      EPSIL=R*ABS(Y(I)-YD(I))
      GO TO (3301,3307),M3
3301 IF(Y(I))3650,3307,3650
3650 EPSIL=EPSIL/(ABS(Y(I)))
3307 IF(SSE-EPSIL)3032,3033,3033
3032 SSE=EPSIL
3033 CONTINUE
      IF(ERMAX-SSE)3034,3034,3035
3034 IF(ABS(DT)-STMIN)42,42,4340
3035 IF(SSE-ERMIN)3036,42,42
3036 IF(STMAX-ABS(DT))42,42,5360
4340 LL=1
      MM=1
      GO TO (4560,4660),M4
4560 WRITE(6,800)SSE,T,DT
      DT=DT*FACT
      WRITE(6,820)XV(1),DT
      GO TO 1001
4660 DT=DT*FACT
      GO TO 1001

```

RKAM 146
 RKAM 147
 RKAM 148
 RKAM 149
 RKAM 150
 RKAM 151
 RKAM 152
 RKAM 153
 RKAM 154
 RKAM 155
 RKAM 156
 RKAM 157
 RKAM 158
 RKAM 159
 RKAM 160
 RKAM 161
 RKAM 162
 RKAM 163
 RKAM 164
 RKAM 165
 RKAM 166
 RKAM 167
 RKAM 168
 RKAM 169
 RKAM 170
 RKAM 171
 RKAM 172
 RKAM 173
 RKAM 174
 RKAM 176
 RKAM 177
 RKAM 178
 RKAM 179
 RKAM 180
 RKAM 181
 RKAM 182
 RKAM 184
 RKAM 185
 RKAM 186
 RKAM 187
 RKAM 188
 RKAM 189
 RKAM 190
 RKAM 191
 RKAM 192
 RKAM 193

| | | | |
|-----|--|------|-----|
| 701 | TZ=ABS(DT)*ZCT | RKAM | 242 |
| 700 | GO TO (700,750,750),MCDE | RKAM | 243 |
| | KT=4 | RKAM | 244 |
| 750 | GO TO 430 | RKAM | 245 |
| 430 | KT=1 | RKAM | 246 |
| | SPACE=A(KM)-XV(KT) | RKAM | 247 |
| | Z=ABS(SPACE) | RKAM | 248 |
| 437 | IF(Z-TZ)437,437,436 | RKAM | 249 |
| 403 | IF(SPACE)413,403,413 | RKAM | 250 |
| | KTM=(KT-1)*NN | RKAM | 251 |
| 404 | DO 404 I=1,NN | RKAM | 252 |
| | Y(I)=UY(KTM+I) | RKAM | 253 |
| | GO TO 443 | RKAM | 254 |
| 436 | KT=KT+1 | RKAM | 255 |
| | IF(KT-5)430,430,790 | RKAM | 256 |
| 413 | KTM=(KT-1)*NN | RKAM | 257 |
| | DO 438 I=1,NN | RKAM | 258 |
| 438 | DY(I)=VF(I+KTM) | RKAM | 259 |
| | KI=-NN | RKAM | 260 |
| | DO 439 K=1,4 | RKAM | 261 |
| | KI=KI+NN | RKAM | 262 |
| | DO 440 I=1,NN | RKAM | 263 |
| | IK=KI+I | RKAM | 264 |
| | YD(IK)=SPACE*DY(I) | RKAM | 265 |
| 440 | Y(I)=UY(I+KTM)+BET(K)*YD(IK) | RKAM | 266 |
| | T=BET(K)*SPACE+XV(KT) | RKAM | 267 |
| 439 | CALL DERFUN(Y,DY,T) | RKAM | 268 |
| | DO 442 I=1,NN | RKAM | 269 |
| | I2=I+NN | RKAM | 270 |
| | I3=I2+NN | RKAM | 271 |
| | I4=I3+NN | RKAM | 272 |
| 442 | Y(I)=UY(KTM+I)+(YD(I2)+YD(I3))+YD(I4))*F16TH | RKAM | 273 |
| 443 | WRITE(6,140)A(KM) | RKAM | 274 |
| | WRITE(6,IOW)(I,Y(I),I=1,NN) | RKAM | 275 |
| | | RKAM | 276 |
| | | RKAM | 285 |
| | U=Y(I)*Y(I)*C.01 | | |
| | DO 948 I=2,201 | | |
| | U=U+C.01*Y(I)*Y(I) | | |
| 948 | CONTINUE | RKAM | 288 |
| 860 | WRITE(6,860)U | | |
| | FORMAT(3H U=,1PE10.3) | | |
| | DX=0.C1 | | |
| | X=0. | | |
| | DO 947 I=1,101 | RKAM | 278 |
| | YX(I,1)=X | RKAM | 280 |
| | X=X+DX | RKAM | 281 |
| | YX(I,2)=Y(2*I-1) | RKAM | 282 |
| 947 | CONTINUE | RKAM | 283 |

C


```

C
C
C
E3=ABS(A(KM)-B(KM))
IF(D(KM))714,713,714
713 E3=ABS(DT*0.5)
714 ABM=E3
GO TO (700,750,750),MCDE
790 E=ABS(XV(J1)-OMEGA)
706 IF(E-EPM)706,648,648
706 EPM=E
GO TO 142

C
C
C
WRITE OUTPUT AT EACH MESHPCINT*****
602 E=ABS(OMEGA-XV(J1))
672 IF(E-EPM)672,672,648
672 EPM=E
600 GO TO (600,650,650),MCDE
600 KT=4
650 GO TO 630
650 KT=1
630 E=ABS(OMEGA-XV(KT))
647 IF(E-E3)647,645,645
647 E3=E
KTM=(K-1)*NN
DO 666 I=1,NN
Y(I)=UY(KTM+I)
CALL DERFUN(Y,DY,T)
WRITE(6,140)XV(KT)
IF(LT-1)733,733,645
645 KT=KT+1
648 IF(KT-5)630,630,142
648 IF(IYVSX)651,651,649
649 CALL VPLOT(YX,JXY,IND,100,NYX,0.0,0.0,C,0.0,0.0)
651 IF(IYVSY)655,655,653
653 CALL VPLOT(YY,MYJ,JND,100,NYY,0.0,0.0,C,0.0,0.0)
655 WRITE(6,220)
RETURN
10 FORMAT(1X,20A4)
20 FORMAT(16I5)
30 FORMAT(20A4)
40 FORMAT(//21H INPUT-OUTPUT FORMATS/1X,10A4,10A4)
50 FORMAT(17H PRINTER PLOT OF ,I2,38H VARIABLE(S) VS. INDEPENDENT VARRKAM
* IARLE./22H VARIABLES PLOTTED ARE,3(A3,I2,1H))
60 FORMAT(/9H ALPHA = ,1PE12.5,M3 M4 M5 M6 IYVSX IYVSY)
70 FORMAT(/48H
80 FORMAT(16I6)
90 FORMAT(71H INITIAL STEP SIZE/72H MAX. ALLOW. MIN. ALLOW. MAX. ALLOW. MIN. ARKAM
*LOW. STEP SINGLE STEP SINGLE STEP ABSOLUTERKAM

```

RKAM 333
 RKAM 334
 RKAM 335
 RKAM 336
 RKAM 337
 RKAM 338
 RKAM 339
 RKAM 340
 RKAM 341
 RKAM 342
 RKAM 343
 RKAM 344
 RKAM 345
 RKAM 346
 RKAM 347
 RKAM 348
 RKAM 349
 RKAM 350
 RKAM 351
 RKAM 352
 RKAM 353
 RKAM 354
 RKAM 355
 RKAM 356
 RKAM 357
 RKAM 358
 RKAM 359
 RKAM 360
 RKAM 361
 RKAM 362
 RKAM 363
 RKAM 364
 RKAM 365
 RKAM 366
 RKAM 367
 RKAM 368
 RKAM 369
 RKAM 370
 RKAM 371
 RKAM 372
 RKAM 374
 RKAM 375
 RKAM 376
 RKAM 378
 RKAM 379
 RKAM 380


```

* * STEP SIZE REDUCTION/69H SIZE ERROR ERROR
100 FORMAT(40X,28H WRITEING AT EACH MESH POINT//)
110 FORMAT(39H STARTING VALUES OF DEPENDENT VARIABLES//)
120 FORMAT(1P6E12.4)
130 FORMAT(12H INDEPENDENT,8X,32HDEPENDENT VARIABLE NO. AND VALUE/
*10H VARIABLE//)
140 FORMAT(1P1E12.4)
150 FORMAT(23H PRINTER PHASE PLOT OF ,2(A3,I2,8H) VS. Y(,I2,1H))
220 FORMAT(1H1)
400 FORMAT(3CX,18HWRITEING INTERVAL,I3//10H START AT ,1P1E14.6,5X, 17H
* INCREMENTING BY ,1E14.6,5X,15H AND FINISH AT ,1E14.6//)
800 FORMAT(/18H MAXIMUM ERROR OF ,1P1E12.5,4H AT ,1P1E12.5,19H WITH STRKAM
*EP SIZE CF ,1P1E12.5)
820 FORMAT( 27H ERROR TOO LARGE, BEGIN AT ,1P1E12.5,19H WITH STEP SIZRRKAM
*E OF ,1P1E12.5//)
840 FORMAT( 32H ERROR TOO SMALL. NEW STEP SIZE ,1P1E12.5//)
END

```

RKAM 381
 RKAM 382
 RKAM 383
 RKAM 384
 RKAM 385
 RKAM 386
 RKAM 387
 RKAM 388
 RKAM 389
 RKAM 390
 RKAM 391
 RKAM 392
 RKAM 393
 RKAM 394
 RKAM 395
 RKAM 396
 RKAM 397
 RKAM 398

```

SUBROUTINE VFLOT(XY,JXY,N,NDIM,NCUR,ISCALE,XL,XU,YL,YU)
DIMENSION IGRID(101),XS(11),YS(17),ICHAR(7),XY(1),JXY(1)
DATA ICHAR/1H+,1H*,1H-,1H$,1H.,1H /
XMIN=XL
XMAX=XU
YMIN=YL
YMAX=YU
IF(ISCALE.NE.0)GO TO 32
XMAX=-1.0E 20
XMIN=-XMAX
YMAX=XMAX
YMIN=XMIN
J2=0
DO 31 J=1,NCUR
J2=J2+2
J1X=(JXY(J2-1)-1)*NDIM
J1Y=(JXY(J2)-1)*NDIM
DO 31 I=1,N
IJX=J1X+I
IJY=J1Y+I
IF(XY(IJX).GT.XMAX)XMAX=XY(IJX)
IF(XY(IJX).LT.XMIN)XMIN=XY(IJX)
IF(XY(IJY).GT.YMAX)YMAX=XY(IJY)
IF(XY(IJY).LT.YMIN)YMIN=XY(IJY)
31 CONTINUE
32 XR=XMAX-XMIN

```

000
 001
 002
 003
 004
 005
 006
 007
 008
 009
 010
 011
 012
 013
 014
 015
 016
 017
 018
 019
 020
 021
 022
 023
 024
 025

```

026 IF(XR.EQ.C.0)XR=1.0E-20
027 YR=YMAX-YMIN
028 IF(YR.EQ.0.0)YR=1.0E-20
029 XT=XMAX*XMIN
030 YT=YMIN*YMAX
031 IF(XT.LT.0.0)YAX=100.0*(-XMIN)/XR+1.5
032 IF(YT.LE.0.0)YAX=64.0*YMAX/YR+1.5
033 XINCR=XR/10.C
034 YINCR=YR/16.C
035 XS(1)=XMIN
036 YS(1)=YMAX
037 DO 46 I=2,11
038 XS(I)=XS(I-1)+XINCR
039 DO 47 I=2,17
040 YS(I)=YS(I-1)-YINCR
041 WRITE(6,10)(XS(I),I=1,11)
042 II=1
043 KK=0
044 DO 146 LINE=1,65
045 DO 101 J=1,101
046 IGRID(J)=ICCHAR(7)
047 IF(YT.GT.0.0)GO TO 109
048 IF(LINE.NE.IXAX)GO TO 109
049 DO 105 J=1,101
050 IGRID(J)=ICCHAR(6)
051 IF(XT.LT.0.0)IGRID(IYAX)=ICCHAR(6)
052 J2=0
053 DO 125 J=1,NCUR
054 JC=MOD(J,5)
055 J2=J2+2
056 JIX=(JXY(J2-1)-1)*NDIM
057 JIY=(JXY(J2)-1)*NDIM
058 DO 125 I=1,N
059 IJX=JIY+I
060 IJY=JIY+I
061 IPTY=64.C*(YMAX-XY(IJY))/YR+1.5
063 IF(IPTY.GT.65)IPTY=65
064 IF(IPTY.LT.1)IPTY=1
065 IF(IPTY.NE.LINE)GO TO 125
066 IPTX=100.0*(XY(IJX)-XMIN)/XR+1.5
068 IF(IPTX.LT.1)IPTX=1
069 IF(IPTX.GT.101)IPTX=101
070 IF(JC.NE.0)GC TO 119
071 IGRID(IPTX)=ICCHAR(5)
072 GO TO 125
073 IGRID(IPTX)=ICCHAR(JC)
115 CONTINUE
125 IF(KK.GT.0)GC TO 134

```

074
075
076
077
078
079
080
081
082
083
084
085
086
087
088

```

WRITE(6,20)YS(II),(IGRID(I),I=1,101),YS(II)
II=II+1
GO TO 135
134 WRITE(6,30)(IGRID(I),I=1,101)
135 KK=KK+1
IF(KK.NE.4)GC TO 146
KK=0
146 CONTINUE
WRITE(6,40)(XS(I),I=1,11)
RETURN
10 FORMAT(1H1,1PE15.2,10E10.2/10X,1H*,20(5H+*****),2H+*)
20 FORMAT(1PE10.2,1H+,101A1,1H+,E9.2)
30 FORMAT(10X,1H*,101A1,1H*)
40 FORMAT(10X,1H*,20(5H+*****),2H+*/1PE16.2,10E10.2)
END

```

```

-----
BLOCK DATA
-----
THIS PROGRAM IS A SIMULATION OF THE 'WIDROW'
FILTER USING THE IBM 2250 GRAPHIC DISPLAY UNIT.
NO INPUT DATA CARDS ARE REQUIRED. FILTER PARAMETERS
MAY BE CHANGED FROM THE 2250. THE ONLY INPUT THAT
IS REQUIRED IS A 9-TRACK, STANDARD LABEL, 800 BPI TAPE
ON WHICH IS THE ACTUAL OR SIMULATED SIGNAL IN DIGITAL
FORM WHICH WOULD APPEAR AT THE OUTPUT OF THE ARRAY
SENSORS.
-----
BLOCK DATA ESTABLISHES NOMINAL INITIAL VALUES
FOR VARIABLES COMMON TO THE MAIN PROGRAM AND THE
SUBROUTINES. NOMINAL VALUES FOR MOST INPUT PARA-
METERS ARE LATER DISPLAYED ON THE 2250 CRT DISPLAY
UNIT AND REPLACED WITH THE DESIRED VALUES FOR THE
NUMERIC KEYBOARD.
-----
COMMON/VALUES/A,D,X(11,48),W(11,11),DESIG,SIGOUT,SIG,
*DT,SPACE,FO,BW,C,DF,KS,ANG,T1,IT1,IT2,IT3,IT4,IT5,LAB,
*ANGL,FREQ,XP(128),XP2(505),YP2(505),TEMP(15),S(64),
*ANGOLD,X,Y(256,4),JXY(4)
COMMON/SIZES/NE,ND,NIT,NCPLX,NF,NA,NT,NN,KDEL,INV(64),
*IT,N1,NNJ,ICCODE2,MP,NP,NAL,NF1,NULL(1)
COMMON/IPLOTS/IPL0T1,IPL0T2,IPL0T3,IPL0T4,IATL,ITEXT1,
*ITEXT2,IATN1,IKEY1
-----

```

```
COMMON/SCALES/XMIN,XMAX,YMAX,FMIN,FMAX,YMIN,ZERO,TEN
COMPLEX A(256),D(256)
```

```
ENTER NOMINAL INITIAL VALUES FOR VARIABLES.
```

```
NE=NUMBER OF ELEMENTS,
ND=NUMBER OF DELAY ELEMENTS
NT=NUMBER OF ITERATIONS OF THE FILTER
NT=NUMBER OF SAMPLING TIME DELAYS PER DELAY ELEMENT
FO=SIGNAL CENTER FREQUENCY
BW=SIGNAL BANDWIDTH OF DESIRED SIGNAL
DT=SAMPLING TIME INTERVAL
REAL KS/C,COO1/,FO/500./,BW/200./,C/1500./,SPACE/1.5/,
*TI/,CO/,ANG/0.0/,DT/1.000E-04/,ANGOLD/225./
REAL ANGL(4)/0.0,30.0,-45.0,88.8/
REAL FREQ(4)/500.0,C,0.800,0.300,0/
DATA NE/1/,ND/2/,NT/49950/,NP/300/,MP/128/,IT/0/,
*N1/1/,NT1/1/,NA/1/,NF/1/,NA1/1/,NF1/1/
DATA XMIN/0.0/,XMAX/1000.0/,YMAX/20.0/,FMIN/0.0/,
*FMAX/1000.0/,YMIN/-90.0/,ZERO/0.0/,TEN/10.0/
```

```
INITIALIZE GRAPH TITLES WITH NOMINAL VALUES FOR
PARAMETERS.
```

```
REAL IT1(24)/'ADAPTIVE FILTER FO= 500,BW= 200',
*,,O= 0.0,T= 000OUTPUT SPECTRUM',
* JCHNSCN BOX 70 1 AUG 69'/
REAL IT2(24)/'ADAPTIVE FILTER FO= 500,BW= 200',
*,,O= 0.0,T= 00FREQUENCY RESPCN',
* SE JCHNSCN BOX 70 1 AUG 69'/
REAL IT3(24)/'ADAPTIVE FILTER FO= 500,BW= 200',
*,,F= 500,T= 00DIRECTIVITY PATT',
* ERN JCHNSCN BOX 70 1 AUG 69'/
REAL IT4(4)/' 45.0 ',LAB/' '
REAL IT5(4)/' 500 '
END
```

```
MAIN PROGRAM
```

IN ADDITION TO THE FILTER EQUATIONS, THE MAIN PROGRAM CONTAINS THE PROCEDURES FOR GETTING THE 2250 DISPLAY UNIT ON LINE, DEFINING THE GRAPHIC DATA SETS TO BE DISPLAYED, ALLOWING INPUT/OUTPUT BETWEEN THE DISPLAY UNIT AND THE PROGRAM, AND HANDLING INTERRUPTS FROM THE 2250 PROGRAMMED FUNCTION KEYS.

```
COMMON/VALUES/A,D,X(11,48),W(11,11),DESIG,SIGOUT,SIG,
```



```

*DT,SPACE,FO,BW,C,DF,KS,ANG,T1,IT1,IT2,IT3,IT4,IT5,LAB,
*ANGL(4),FREQ(4),XP(128),XP2(505),YP2(505),TEMP(15),
*S(64),ANGOLD,XY(256,4),JXY(4)
*COMMON/SIZES/NE,ND,NIT,NCPLX,NF,NA,NT,NN,KDEL,INV(64),
*IT,N1,NNJ,ICCODE2,MP,NP,NAI,NFI,NULL(1)
*COMMON/IPLOTS/IPLOT1,IPLOT2,IPLOT3,IPLOT4,IATL,ITEXT1,
*ITEXT2,IATN1,IKEY1
*COMMON/SCALES/XMIN,XMAX,YMAX,FMIN,FMAX,YMIN,ZERO,TEN
COMPLEX A(256),D(256)
REAL IT1(24),IT2(24),IT3(24),IT4(4),IT5(4),LAB,KS
DIMENSION SIGIN(11,11)
DEFINE FILE 11(5,100,E,J3)
NULL(1)=-3
ICCODE=9
ICCODE1=15
ICCODE2=ICCODE1
-----
INITIALIZE GRAPHIC SUBROUTINE PACKAGE, GRAPHIC
DEVICE, AND GRAPHIC DATA SETS.
-----
CALL INGSPL(IPLLOT,NULL)
CALL INDEV(IPLLOT,10,I2250,512)
CALL INGDS(I2250,IPLLOT1)
CALL INGDS(I2250,IPLLOT2)
CALL INGDS(I2250,IPLLOT3)
CALL INGDS(I2250,IPLLOT4,256)
CALL INGDS(I2250,ITEXT1,256)
CALL INGDS(I2250,ITEXT2,256)

SET GRAPHIC AND CHARACTER MODES.

CALL SDATM(ITEXT1,3)
CALL SCHAM(ITEXT1,4)
CALL SCHAM(ITEXT2,4)

SET GRAPHIC DATA SET LIMITS.

CALL SGDSL(IPLLOT1,1.0,1.0,11.0,11.0,0.0,0.0,12.0,12.0)
CALL SGDSL(IPLLOT2,1.0,1.0,11.0,11.0,0.0,0.0,12.0,12.0)
CALL SGDSL(IPLLOT3,0.0,0.0,12.0,12.0,0.0,0.0,12.0,12.0)
CALL SGDSL(IPLLOT4,1.0,1.0,11.0,11.0,0.0,0.0,12.0,12.0)
CALL SGDSL(ITEXT1,1.0,1.0,10.0,10.0,0.0,0.0,12.0,12.0)

SET DATA LIMITS.

CALL SDATL(IPLLOT1,XMIN,ZERC,XMAX,YMAX)
CALL SDATL(IPLLOT2,FMIN,YMIN,FMAX,TEN)
CALL SDATL(IPLLOT3,-1.2,-1.2,1.2,1.2)

```

```

CC C      CALL SDATL(IPLOT4,0.0,0.0,0.0,10.0,10.0,10.0)
CC C      CALL SDATL(ITEXT1,0.0,0.0,50,35)
CC C      CALL SDATL(ITEXT2,0.0,0.0,0.0,10.0,10.0)
CC C      SET SCISSCRING OPTIONS.
CC C      CALL SSCIS(ITEXT1,3)
CC C      CALL SSCIS(ITEXT2,3)
CC C      ENABLE ATTENTIONS FRMC SPECIFIED PROGRAMMED FUNCTION
CC C      KEYS AND FROM THE KEYBOARD 'END' KEY.
CC C      CALL CRATL(I2250,IATL)
CC C      CALL ENATN(IATL,1,-3,8,9,15,21,22,27,-32)
CC C      GENERATE A 10 INCH BY 10 INCH GRID.
CC C      DO 86 I=1,11
CC C      X1=I-1
CC C      CALL STPCS(IPLOT4,X1,0.0)
CC C      CALL PLINE(IPLOT4,X1,10.0)
CC C      Y=X1
CC C      CALL STPOS(IPLOT4,0.0,Y)
CC C      CALL PLINE(IPLOT4,10.0,Y)
CC C      86 ----- ENTER DESIRED FILTER PARAMETERS FOR THIS RUN. -----
CC C      -----
CC C      DISPLAY NCMINAL GRAPH TITLE.
CC C      87 CALL PTEXT(ITEXT1,IT1(1),20,NULL,NULL,1,C,31)
CC C      CALL PTEXT(ITEXT1,IT1(6),28,1)
CC C      CALL PTEXT(ITEXT1,IT1(13),36,11,NULL,1,0,30)
CC C      CALL PTEXT(ITEXT1,IT1(22),12,2)
CC C      CALL EXEC(ITEXT1)
CC C      INSERT CURSOR INTO DISPLAY SC THAT NEW DATA MAY BE
CC C      ENTERED.
CC C      CALL ICURS(ITEXT1,1)
CC C      WAIT UNTIL ATTENTION FRMC 'END' KEY INDICATES THAT
CC C      THE DATA HAS BEEN ENTERED.
CC C      CALL RQATN(IATL,IATN5,2,NULL,32)
CC C      MOVE CURSOR TO NEXT LINE AND WAIT FOR AN ATTENTION
CC C      FROM THE 'END' KEY.

```



```

C      CALL ICURS(ITEXT1,2)
CC     CALL RQATN(IATL,IATN5,2,NULL,32)
CC
CC     READ NEW PARAMETERS INTO GRAPH TITLE.
CC
CC     CALL GSPRD(ITEXT1,IT1(6),28,1,NULL,1)
CC     CALL GSPRD(ITEXT1,IT1(22),12,1,NULL,2)
CC     CALL GSPRD(ITEXT1,IT1(13),8,1,NULL,11)
CC
CC     CHANGE CORRESPONDING PARAMETERS IN THE OTHER TWO
CC     GRAPH TITLES.
CC
88     DO 88 I=6,8,2
      IT2(I)=IT1(I)
      IT3(I)=IT1(I)
89     DO 89 I=22,24
      IT2(I)=IT1(I)
89     IT3(I)=IT1(I)
CC
CC     CHANGE NOMINAL INPUT VALUES INTO 'A' FORMAT.
CC
CC     WRITE(11,1,1000) NE,ND,SPACE,C,KS,FO,BW,ANG,NT
CC     READ(11,1,1001) TEMP
CC
CC     DISPLAY REQUEST FOR INPUT DATA USING NOMINAL VALUES
CC
      CALL PTEXT(ITEXT1,'# CF ELEMENTS'          ='',22,NULL,
*NULL,1,1,26)
      CALL PTEXT(ITEXT1,TEMP(1),4,3)
      CALL PTEXT(ITEXT1,'# CF DELAYS'            ='',22,NULL,
*NULL,1,1,24)
      CALL PTEXT(ITEXT1,TEMP(2),4,4)              DT *,22,NULL,
      CALL PTEXT(ITEXT1,'EACH DELAY =           ='',22,NULL,
*NULL,1,1,22)
      CALL PTEXT(ITEXT1,TEMP(15),4,12)
      CALL PTEXT(ITEXT1,'ELEMENT SPACING'        ='',22,NULL,
*NULL,1,1,20)
      CALL PTEXT(ITEXT1,TEMP(3),8,5)              ='',22,NULL,
      CALL PTEXT(ITEXT1,'SPEED OF SOUND'         ='',22,NULL,
*NULL,1,1,18)
      CALL PTEXT(ITEXT1,TEMP(5),8,6)
      CALL PTEXT(ITEXT1,'CONVERGENCE CONSTANT'   ='',22,NULL,
*NULL,1,1,16)
      CALL PTEXT(ITEXT1,TEMP(7),8,7)
      CALL PTEXT(ITEXT1,'CENTER FREQUENCY'       ='',22,NULL,
*NULL,1,1,14)
      CALL PTEXT(ITEXT1,TEMP(9),8,8)

```

```

CALL PTEXT(I TEXT1, 'BANDWIDTH
*NULL,1,1,12)
CALL PTEXT(I TEXT1, TEMP(11), 8, 9)
CALL PTEXT(I TEXT1, 'DESIRED LOOK ANGLE
*NULL,1,1,10)
CALL PTEXT(I TEXT1, TEMP(13), 8, 10)
CALL EXEC(I TEXT1)
CALL ICURS(I TEXT1, 3)
CALL RQATN(I ATL, I ATN5, 2, NULL, 32)
CALL ICURS(I TEXT1, 4)
CALL RQATN(I ATL, I ATN5, 2, NULL, 32)
CALL ICURS(I TEXT1, 12)
CALL RQATN(I ATL, I ATN5, 2, NULL, 32)
DO 90 I=5,10
CALL ICURS(I TEXT1, I)
90 CALL RQATN(I ATL, I ATN5, 2, NULL, 32)

READ INPUT VALUES IN 'A' FORMAT.

CALL GSPRD(I TEXT1, TEMP(1), 4, 1, NULL, 3)
CALL GSPRD(I TEXT1, TEMP(2), 4, 1, NULL, 4)
CALL GSPRD(I TEXT1, TEMP(15), 4, 1, NULL, 12)
IND=3
DO 91 I=5,10
CALL GSPRD(I TEXT1, TEMP(IND), 8, 1, ITRM, I)
91 IND=IND+2

CHANGE VALUES INTO PROPER FORMAT.

WRITE(11,1,1001) TEMP
READ(11,1,1000) NE,ND,SPACE,C,KS,FO,8W,ANG,NT

WRITE FILTER PARAMETERS CN PRINTED OUTPUT.

WRITE(6,2000) IT,NE,SPACE
WRITE(6,2001) ND,NT,DT
WRITE(6,2002) C

DISPLAY REQUEST FOR INITIAL PARAMETERS FOR THE
'FREQUENCY RESPONSE' AND 'DIRECTIVITY PATTERN'
PLOTS FOR THIS RUN.

CALL PLTDAT(692)

GENERATE AND DISPLAY 'ANGLE' AND 'TIME', USING
NOMINAL VALUES.

92 CALL PTEXT(I TEXT2, 'ANGLE =' , 7, NULL, NULL, 1, 1, 0, 0, 3)

```

```

C      CALL PTEXT(ITEXT2,TEMP(13),8,1)
C      CALL PTEXT(ITEXT2,TIME=,6,NULL,1,6.0,0.3)
C      CALL PTEXT(ITEXT2,'0000',4,2,IKEY1,1,7.5,0.3)
C      CALL EXEC(ITEXT2)

C      DISPLAY GRID LINES.

C      CALL EXEC(IPL0T4)
C-----
C      INITIALIZE FILTER EQUATIONS
C-----
C      IFF=0
C      IP=1
C      NNJ=ND+1
C      NJ=NNJ*NT
C      L=NJ
C      NJP=NJ+1
C      KI=NJP
C      NEO2=NE/2

C      SYNTHESIZE DESIRED SIGNAL.

C      CALL SIGREF(87)

C      CALCULATE TIME DELAY BETWEEN ELEMENTS REQUIRED TO
C      OBTAIN THE DESIRED LOOK ANGLE.

C      7 CALL DIRECT

C      CALCULATE X AXIS ARRAY FOR 'OUTPUT SPECTRUM' PLOT.

C      DO 8 J=1,MF
C      8 XP(J)=(J-1)*DF

C      INITIALIZE SIMULATED DELAY LINE.

C      SIGMA=0.7071C7
C      IX=1172903975
C      DO 9 J=1,NJ
C      LL=NJP-J
C      READ(9) (X(I,LL),I=1,11)
C      DO 9 I=1,11
C      9 CALL GAUSS(IX,SIGMA ,0.0,X(I,LL))

C-----
C      START OF FILTER EQUATIONS
C-----
C      DO 5 M=1,NIT
C      READ(9) (X(I,L),I=1,11)

```



```

C-----
C          BRANCHING AND CONTROL EQUATIONS
C-----
C- L1 -- HAVE 'NP' ADDITIONAL ITERATIONS OF THE FILTER
C          EQUATIONS BEEN PERFORMED ? ----->
C
C          6 IF(MOD(M,NP)) 5,20,5
C----->
C          NO. CONTINUE ITERATING FILTER EQUATIONS.
C----->
C          YES. UPDATE 'TIME' DISPLAY, THEN GO TO L2. ->
C
C          20 IT=M/10
C             IFF=0
C             IP=0
C             WRITE(11,2,1002) IT
C             READ(11,2,1003) IT
C             CALL RESET(ITEXT2,2,IKEY1)
C             CALL PTEXT(ITEXT2,IT,4,2,IKEY1,1,7.5,0.3)
C             CALL EXEC(ITEXT2)
C
C- L2 -- HAS THERE BEEN AN ATTENTION FROM PFK-8, PFK-9,
C          PFK-15, PFK-21, OR PFK-27 ? ----->
C
C          CALL RQATN(IATL,IATN1,1,NULL,8,9,15,21,27)
C----->
C          NO. GO TO L3. ----->
C
C          IF(IATN1.LT.5) GC TO 103
C----->
C          YES. IF ATTENTION WAS FROM : ----->
C
C          ICODE=IATN1
C          IF(ICODE-5) 100,101,102
C----->
C          PFK-8. REMOVE GRID LINES FROM DISPLAY,
C          THEN GO TO L3. ----->
C
C          100 CALL OMIT(IPLOT4)
C             GO TO 103
C----->
C          PFK-9. DISPLAY GRID LINES, THEN GO TO
C          L3. ----->
C
C          101 CALL INCL(IPLOT4)
C             GO TO 103
C
C

```



```

C- L6.2- HAS LAST 'FREQUENCY RESPONSE' GRAPH FOR THIS TIME ----->
C- STEP BEEN PLOTTED ? ----->
C----->
C-----> YES. GO TO L6. ----->
C----->
C-----> NO. CALCULATE, DISPLAY, AND PLOT NEXT GRAPH.
C----->
C 122 NAI=NAI+1
C IF(NAI.LE.NA) GO TO 1116
C- L6 -- TURN ON LIGHT IN PFK-1 AND WAIT UNTIL AN ATTENTION
C OCCURS FROM PFK-1, PFK-2, PFK-3, OR 'END' KEY. ---->
C 113 CALL MLITS(IATL,4,1)
C CALL RQATN(IATL,IATN4,2,NULL,1,2,3,32)
C-----> IF ATTENTION IS FROM : ----->
C----->
C IF(IATN4.EQ.32) GO TO 118
C IF(IATN4-2) 114,116,115
C----->
C-----> PFK-1. IF GRAPH PRESENTLY DISPLAYED IS
C-----> 'OUTPUT SPECTRUM', GENERATE DATA
C-----> FOR CALCOMP PLOT OF SAME.
C-----> OTHERWISE, CALCULATE AND DISPLAY
C-----> 'OUTPUT SPECTRUM', THEN GENERATE
C-----> DATA FOR CALCOMP PLOT OF SAME.
C-----> RETURN TO L6. ----->
C 114 IF(ICODE2.EQ.15) GO TO 1124
C 1114 CALL SPECTR(81124)
C 1124 CALL PSPECT(81113)
C----->
C-----> PFK-3. IF GRAPH PRESENTLY DISPLAYED IS
C-----> 'DIRECTIVITY PATTERN', GENERATE
C-----> DATA FOR CALCOMP PLOT OF SAME.
C-----> OTHERWISE, CALCULATE AND DISPLAY
C-----> 'DIRECTIVITY PATTERN', THEN
C-----> GENERATE DATA FOR CALCOMP PLOT
C-----> OF SAME. RETURN TO L6.1. ----->
C 115 IF(ICODE2.EQ.27) GO TO 1125
C 1115 CALL PATTERN(81125)
C 1125 CALL PPATRN(81121)
C----->
C-----> PFK-2. IF GRAPH PRESENTLY DISPLAYED IS
C-----> 'FREQUENCY RESPONSE', GENERATE

```

```

C C C C C
DATA FOR CALCOMP PLOT OF SAME.
C OTHERWISE, CALCULATE AND DISPLAY
'FREQUENCY RESPONSE', THEN GENER-
ATE DATA FOR CALCOMP PLOT OF
SAME. RETURN TO L6.2. ----->
C C C C C
116 IF(ICODE2.EQ.21) GO TO 1126
1116 CALL FREQCY(61126)
1126 CALL PREQ(6122)
C----->
C C C C C
'END'
KEY.
TURN LIGHT IN PFK-1 OFF, THEN
CONTINUE ITERATING FILTER EQUA-
TIONS.
C C C C C
118 CALL MLITS(IATL,2)
NFI=1
NAI=1
5 IP=IP+1
117 CONTINUE
117 REWIND 5
C C C C C
C- L7 -- HAS THERE BEEN AN ATTENTION FROM THE 'END' KEY ? -->
C C C C C
CALL RGATN(IATL,IATN9,1,NULL,32)
IF(IATN9.EQ.32) GO TO 200
C C C C C
C-----> NC. REMOVE GRID AND GRAPH FROM DISPLAY, RESET
PROBLEM AND REQUEST NEW INPUT DATA BY
DISPLAYING PRESENT PARAMETERS.
C C C C C
CALL RESET(ITEXT1)
CALL OMIT(IPLOT4)
N1=1
IT=0
IF(ICODE2-21) 150,151,152
150 CALL RESET(IPLOT1)
GO TO 87
151 CALL RESET(IPLOT2)
GO TO 87
152 CALL RESET(IPLOT3)
GO TO 87
C C C C C
C-----> YES. TERMINATE THE GRAPHIC SUBROUTINE PACKAGE
C C C C C
200 CALL TMGSP(IPLOT)
C-----
C-----
C-----
TERMINATE SIMULATION
C-----

```



```

*0.3,4.0)
CALL PTEXT(I TEXT2, TEMP(9),8,3)
CALL PTEXT(I TEXT2, IT1(6),4,4)
CALL PTEXT(I TEXT2, BANDWIDTH =, 11, NULL, NULL, 1, 0.3, 3.5)
CALL PTEXT(I TEXT2, TEMP(11),8,5)
CALL PTEXT(I TEXT2, IT1(8),4,6)
CALL PTEXT(I TEXT2, CCNVERGENCE CONSTANT =, 22, NULL, NULL,
*1,0.3,3.0)
CALL PTEXT(I TEXT2, TEMP(7),8,7)
CALL EXEC(I TEXT2)
CALL ICURS(I TEXT2, 3)
CALL RQATN(IATL, IATN6, 2, NULL, 32)
CALL ICURS(I TEXT2, 5)
CALL RQATN(IATL, IATN6, 2, NULL, 32)
CALL ICURS(I TEXT2, 7)
CALL RQATN(IATL, IATN6, 2, NULL, 32)
CALL GSPRD(I TEXT2, TEMPI(1), 12, 1, ITERM, 3, NULL, 4)
CALL GSPRD(I TEXT2, TEMPI(4), 12, 1, ITERM, 5, NULL, 6)
CALL RESET(I TEXT2, NULL, IKEY)
CALL MLITS(IATL, 2)
IT1(6)=TEMPI(3)
IT2(6)=IT1(6)
IT3(6)=IT1(6)
IT1(8)=TEMPI(6)
IT2(8)=IT1(8)
IT3(8)=IT1(8)
TEMP(9)=TEMPI(1)
TEMP(10)=TEMPI(2)
TEMP(11)=TEMPI(4)
TEMP(12)=TEMPI(5)
TEMP(7)=TEMPI(7)
TEMP(8)=TEMPI(8)
WRITE(11,2,1000) TEMPI
READ(11,2,1001) FO,BW,KS
IF(ICODE2.GT.21) GC TC 105
CALL INCL(IPL0T4)
CALL CANCEL(2)
NCPLX=256
PI=3.141593
NN=NCPLX
DF=1.0/(DT*NN)
NFO=BW/DF+0.5
RNFO=NFO
DO 1 N=1,NN
1 D(M) =(C.G.O.O)
FMAG=C.500
IF(NFO.NE.O) FMAG=SQRT(FMAG/RNFO)
105

```



```

NL=FO*DT*NN-NFO/2+1.5
NU=NL+NFC
DO 2 K=NL,NU
  D(K) =FMAG
NL=NN+2-NU
NU=NL+NFC
DO 3 K=NL,NU
  D(K) =FMAG
JXY(1)=1
JXY(2)=2
FI=0
DO 31 I=1,128
  XY(I,1)=FI
  FI=FI+DF
31 XY(I,2)=D(I)
  CALL VPLOT(XY,JXY,128,256,1,0,0,0,0,0,0,0,0)
  CALL SFRT(D,NCPLX,+1.0)
  WRITE(6,2000) IT,FO,Bk
  WRITE(6,2003) KS
  TT=0
DO 33 I=1,256
  XY(I,1)=TT
  TT=TT+DT
  XY(I,2)=D(I)
33 CONTINUE
15 CALL VPLOT(XY,JXY,256,256,1,0,0,0,0,0,0,0,0)
  FORMAT(5(I4,1P2E10.2))
  RETURN 1
11 FORMAT(4(I1X,'D(',I4,')= ',2F9.5,4X))
12 FORMAT(1H1)
13 FORMAT(12)
1000 FORMAT(8A4)
1001 FORMAT(F8.2,'X,F8.2,4X,F8.6)
2000 *F8.2,' BANDWIDTH = ',F8.2)
2003 *F8.2,' CCNVERGENCE CONSTANT = ',F9.6)
      END

```

~~~~~

~~~~~

~~~~~

REMOVE  
THESE  
CARDS  
WHEN  
CURS-  
FIXED

```

DO 4 J=1,NJ
W(I,J)=WINIT
WING=C*DT/SPACE
KDEL=SIN(ANG/57.29578)/ARG+SIGN(0.5,ANG)
ANG=ARCSIN(KDEL*ARG)*57.29578
WRITE(6,2000) IT,ANG
WRITE(11,3,1001) ANG
READ(11,3,1000) TEMP2
TEMP(13)=TEMP2(1)
TEMP(14)=TEMP2(2)
IT1(10)=ABS(ANG)
WRITE(11,3,1002) IT1(10)
READ(11,3,1003) IT1(10)
IF(ANG) 1,2,2
1 IT1(9)=IT7
2 GO TO 3
3 IT1(5)=IT6
CALL RESET(IITEXT2)
CALL PTEXT(IITEXT2, 'ANGLE =', 7, NULL, NULL, 1, 1.0, 0.3)
CALL PTEXT(IITEXT2, 'TEMP(13)', 8, 1)
CALL PTEXT(IITEXT2, 'TIME =', 6, NULL, NULL, 1, 6.0, 0.3)
CALL PTEXT(IITEXT2, 'I1,4,2,IKEY1,1,7.5,0.3')
CALL EXEC(IITEXT2)
RETURN
FORMAT(2A4)
1000 FORMAT(F8.2)
1001 FORMAT(F4.1)
1002 FORMAT(A4)
1003 FORMAT('0', ' T =', I5, ' DESIRED LOOK ANGLE = ', F6.2,
* DEGREES')
END

```

SUBROUTINE PLTDAT(\*)

SUBROUTINE PLTDAT READS THE PARAMETERS FOR WHICH THE CALCOMP PLOTS ARE TO BE MADE. AT ANY ONE STEP IN TIME, UP TO FOUR FREQUENCY RESPONSE, AND FOUR DIRECTIVITY PATTERN, GRAPHS CAN BE PLOTTED. THE GRAPH WHICH WILL BE DISPLAYED ON THE 2250 DISPLAY, WILL BE CALCULATED FOR THAT PARAMETER WHICH APPEARS FIRST IN THE PARAMETER LIST FOR THE CORRESPONDING GRAPH. WHILE CALCOMP PLOTS ARE BEING MADE, THE GRAPH BEING PLOTTED WILL BE DISPLAYED SIMULTANEOUSLY ON

۷۷۷

2 95



```

CALL GSPRD(I,TEXT1,NAN,4,1,IRRD,1)
CALL GSPRD(I,TEXT1,ANGLE,32,1,IRRD,2)
CALL GSPRD(I,TEXT1,NFR,4,1,IRRD,3)
CALL GSPRD(I,TEXT1,FREQ,32,1,IRRD,4)
CALL RESET(I,TEXT1)
CALL MLITS(IATL,2)
WRITE(11,4,1001) NAN,ANGLE
READ(11,4,1000) NA,ANGL
WRITE(11,5,1001) NFR,FREQ
READ(11,5,1000) NF,FREQ
DO 106 N=1,4
  IT7(N)=FREQ(N)
  IT4(N)=ABS(ANGL(N))
  WRITE(11,4,1002) IT4
  READ(11,4,1003) IT4
  WRITE(11,5,1004) IT7
  READ(11,5,1003) IT5
  IATN1=ICCODE2
  CALL SSCAL
  RETURN 1
1000 FORMAT(I4,4F8.2)
1001 FORMAT(A4,8A4)
1002 FORMAT(4F4.1)
1003 FORMAT(4A4)
1004 END

```

# SUBROUTINE SSCAL

SUBROUTINE SSCAL CALCULATES AND DISPLAYS ON THE 2250 SCALES FOR THE FREQUENCY RESPONSE, AND FOR THE OUTPUT SPECTRUM, GRAPHS. ALL SCALES ARE DISPLAYED IN INTEGER NUMBERS, HOWEVER, THE MAXIMUM ORDINATE FOR THE OUTPUT SPECTRUM, PLOT IS MULTIPLIED INTERNALLY BY A SCALE FACTOR TO ACHIEVE THE DESIRED RANGE.

```

COMMON/VALUES/A,D,X(11,48),W(11,11),DESIG,SIGOUT,SIG,
*DT,SPACE,FO,BW,C,DF,KS,ANG,T1,IT1,IT2,IT3,IT4,IT5,LAB,
*ANGL(4),FREQ(4),XP(128),XP2(505),YP2(505),TEMP(15),
*S(64),ANGOLD,XY(256,4),JXY(4)
COMMON/SIZES/NE,ND,NIT,NCPLX,NF,NA,NT,NN,KDEL,INV(64),
*IT,N1,NNJ,ICCODE2,MP,NP,NAI,NF1, NULL(1)

```



```

COMMON/IPLOTS/IPLT1,IPLCT2,IPLCT3,IPLCT4,IATL,ITEXT1,
*ITEXT2,IATN1,IKEY1
COMMON/SCALES/XMIN,XMAX,YMAX,FMIN,FMAX,YMIN,ZERO,TEN
COMPLEX A(256),D(256)
REAL IT1(24),IT2(24),IT3(24),IT4(4),IT5(4),LAB,KS
DIMENSION ITEMP(4)
DEFINE FILE 11(5,100,E,J3)
IF(IATN1-21) 1,2,4
1 ITEMP(1)=YMAX*100.01
  ITEMP(2)=ZEROC
  ITEMP(3)=XMIN
  ITEMP(4)=XMAX
WRITE(11,4,100) ITEMP
READ(11,4,100) ITEMP
GO TO 3
2 ITEMP(1)=TEN
  ITEMP(2)=YMIN
  ITEMP(3)=FMIN
  ITEMP(4)=FMAX
WRITE(11,4,100) ITEMP
READ(11,4,100) ITEMP
3 CALL RESET(ITEXT1, ITEMP(1),4,1,NULL,1,0,32)
  CALL PTEXT(ITEXT1, ITEMP(2),4,2,NULL,1,0,3)
  CALL PTEXT(ITEXT1, ITEMP(3),4,3,NULL,1,3,2)
  CALL PTEXT(ITEXT1, ITEMP(4),4,4,NULL,1,45,2)
  CALL EXEC(ITEXT1)
  CALL INCL(IPLCT4)
GO TO 5
4 CALL RESET(ITEXT1)
  CALL OMIT(IPLCT4)
5 RETURN
ENTRY CSCALE

```

```

-----
CCCCCCCCCCCC
ENTRY CSCALE PERMITS THE SCALES OF THE GRAPHS
ON THE 2250 TO BE CHANGED TO EXPAND OR COMPRESS THE
PLOTS AS DESIRED. THE NEW SCALES ARE READ FROM THE
2250 AND THE DATA LIMITS FOR THE GRAPHIC DATA SET
ARE CHANGED ACCORDINGLY. NOTE. THIS DOES NOT
CHANGE THE SCALING OF THE CALCOMP PLOTS.
-----

```

```

IF(ICODE2.GT.21) GO TO 9
CALL MLITS(IATL,4,31)
IF(ICODE2-21) 101,102,103
101 CALL RESET(IPLCT1)
GO TO 104

```

REMOVE  
THESE  
CARDS

WHEN  
-RCURS-  
FIXED

```

102 CALL RESET(IFLOT2)
103 GO TO 104
104 CALL RESET(IFLOT3)
CALL ICURS(ITEMT1,1)
CALL IQATN(IATL,IATN10,2,NULL,32)
CALL ICURS(ITEMT1,2)
CALL IQATN(IATL,IATN10,2,NULL,32)
CALL ICURS(ITEMT1,3)
CALL IQATN(IATL,IATN10,2,NULL,32)
CALL ICURS(ITEMT1,4)
CALL IQATN(IATL,IATN10,2,NULL,32)
CALL ICURS(ITEMT1)
CALL GSPRD(ITEMT1,ITEMP(3),8,1,IRRD,3,NULL,4)
WRITE(11,5,1C01) ITEMF
IF(ICCDE2.GT.15) GC TC 7
6 READ(11,5,10C2) YMAX,ZERO,XMIN,XMAX
YMAX=YMAX#0.C1
CALL SDATL(IFLOT1,XMIN,ZERC,XMAX,YMAX)
GO TO 8
7 READ(11,5,10C2) TEN,YMIN,FMIN,FMAX
CALL SDATL(IFLOT2,FMIN,YMIN,FMAX,TEN)
CALL MLITS(IATL,2)
8 RETURN
9 FORMAT(4I4)
1000 FORMAT(4A4)
1001 FORMAT(4F4.0)
1002 END

```

# SUBROUTINE SPECTR(\*)

SUBROUTINE SPECTR CALCULATES AND DISPLAYS THE  
'OUTPUT SPECTRUM' GRAPH USING THE FAST FOURIER  
TRANSFORM TO OBTAIN THE FREQUENCY SPECTRUM OF THE  
FILTER OUTPUT SIGNAL.

```

COMMON/VALUES/A,D,X(11,48),W(11,11),DESIG,SIGOUT,SIG,
*DT,SPACE,FO,BW,C,DF,KS,ANG,T1,I1,I2,IT3,IT4,IT5,LAB,
*ANGL(4),FREQ(4),XP(128),XP2(505),YP2(505),TEMP(15),
*SI(64),ANGOLD,XY(256,4),JXY(4)
COMMON/SIZES/NE,ND,NIT,NCPLX,NF,NA,NT,NN,KDEL,INV(64),
*IT,N1,NNJ,ICCDE2,MP,NP,NFI,NFL,NULL(1)
COMMON/IPLOTS/IPL0T1,IPL0T2,IPL0T3,IPL0T4,IATL,ITEMT1,

```



```

-----
SUBROUTINE FREQCY(*)
-----
      SUBROUTINE FREQCY CALCULATES AND DISPLAYS THE
      FREQUENCY RESPONSE, GRAPH. THE FREQUENCY RESPONSE
      OF THE FILTER IS CALCULATED, USING THE VALUES OF THE
      WEIGHTS AT THE TIME THAT THE SUBROUTINE IS CALLED,
      FOR THE ANGLE PARAMETERS LAST READ BY SUBROUTINE
      PLTDAT.

      COMMON/VALUES/A,D,X(11,48),W(11,11),DESIG,SIGOUT,SIG,
      *DT,SPACE,FO,BW,C,DF,KS,ANG,T1,IT1,IT2,IT3,IT4,IT5,LAB,
      *ANGL(4),FREQ(4),XP(128),XP2(505),YP2(505),TEMP(15),
      *S(64),ANGOLD,XY(256,4),JXY(4)
      COMMON/SIZES/NE,ND,NIT,NCPLX,NF,NA,NT,NN,KDEL,INV(64),
      *IT,N1,NNJ,ICDDE2,MP,NP,NAL,NF1,NULL(1)
      COMMON/IPLOTS/IPLOT1,IPLCT2,IPLCT3,IPLCT4,IATL,ITEXT1,
      *ITEXT2,IATN1,IKEY1
      COMMON/SCALES/XMIN,XMAX,YMAX,FMIN,FMAX,YMIN,ZERO,TEN
      COMPLEX A(256),D(256)
      REAL IT1(24),IT2(24),IT3(24),IT4(4),IT5(4),LAB,KS
      DEFINE FILE 11(5,100,E,J3)
      REAL IT6/4H,C=/,IT7/4H,O=-/
      DIMENSION GRF(505,2)
      EQUIVALENCE(GRF(1,1),XP2(1)),(GRF(1,2),YP2(1))
      IF(ICDDE2-21) 101,103,102
      CALL RESET(IPLOT1)
101 GO TO 103
102 CALL RESET(IPLCT3)
103 TWOPI=6.2831853
      DT1=NT*DT
      NEO2=NE/2+1
      LOWF=FMIN+2.
      MAXF=FMAX
      IDF=(MAXF+495-LOWF)/500
      ANGL1=ANGL(NAL)/57.29578
      DELT=SPACE*SIN(ANGL1)/C
      II=0
      AMPMAX=NE
      DO 2 K=LCWF,MAXF,IDF
      OMEGA=TWOPI*K
      PHI=OMEGA*DT1
      PSI=OMEGA*DELT
      AMPL=C.C
      II=II+1
      DO 1 I=1,NE
      FPSI=(I-NEC2)*PSI

```





SUBROUTINE PATERN(\*)

SUBROUTINE PATERN CALCULATES AND DISPLAYS THE DIRECTIONALITY PATTERN GRAPH. THE DIRECTIONALITY PATTERN OF THE FILTER IS CALCULATED, USING THE VALUES OF THE WEIGHTS AT THE TIME THAT THE SUBROUTINE IS CALLED, FOR THE FREQUENCY PARAMETERS LAST READ BY SUBROUTINE PLTDAT. POINTS IN THE CALCULATION OF THE DESIRED AN ARROW WHICH OF THE FILTER. THE BASE OF THE ARROWHEAD INDICATES THE DESIRED MAGNITUDE OF THE DIRECTIONALITY PATTERN IN THE DESIRED LOOK DIRECTION.

```
COMMON/VALUES/A,D,X(11,48),W(11,11),DESIG,SIGOUT,SIG,
*DT,SPACE,FC,BW,C,DF,KS,ANG,T1,IT1,IT2,IT3,IT4,IT5,LAB,
*ANGL(4),FREQ(4),XP(128),XP2(505),YP2(505),TEMP(15),
*S(64),ANGOLD,XY(256,4),JXY(4)
*IT,N1,NNJ,ICCODE2,MP,NP,NCPLX,NF,NA,NT,NN,KDEL,INV(64),
COMMON/IPLCTS/IPLCT1,IPLCT2,IPLCT3,IPLCT4,IATL,ITEXT1,
*ITEXT2,IATN1,IKEY1
COMMON/SCALES/XMIN,XMAX,YMAX,FMIN,FMAX,YMIN,ZERO,TEN
COMPLEX A(256),D(256)
REAL IT1(24),IT2(24),IT3(24),IT4(4),IT5(4),LAB,KS
DIMENSION XXP(64),YYP(64)
DEFINE FILE 11(5,100,E,J3)
DIMENSION TEMP3(2)
IF(ICCODE2-21) 101,102,103
CALL RESET(IPLCT1)
GO TO 103
101 CALL RESET(IPLCT2)
102 CALL RESET(IPLCT2)
103 DT1=NT*DT
NEO2=NE/2+1
IF(ANG.EQ.ANGOLD) GO TO 6
PI=3.1415927
ANGOLD=ANG
LL=53
NPT=181
DRETA=1.745329E-02
LP=LL+7
ANG1=ANG/57.29578
NPT02=NPT/2+1
XS=SIN(ANG1)*0.02
YS=COS(ANG1)*0.02
DO 4 M=1,LL
XXP(M)=W*XS
```

```

4 YYP(M)=N*YS
  XSM=SIN(ANG1-0.26)*0.02
  YSM=COS(ANG1-0.26)*0.02
  XSP=SIN(ANG1+0.26)*0.02
  YSP=COS(ANG1+0.26)*0.02
  DO 5 L=1,3
    LLP=LL+L
    LPM=LP-L
    XXP(LLP)=XXP(LL)-L*XSM
    YYP(LLP)=YYP(LL)-L*YSM
    XXP(LPM)=XXP(LL)-L*XSP
    YYP(LPM)=YYP(LL)-L*YSP
    XXP(LP)=XXP(LL)
    YYP(LP)=YYP(LL)
    OMEGA=2.0*PI*FREQ(NF1)
    PHI=OMEGA*DTI
    AMPMAX=NE
  DO 2 K=1,NPT
    AMPL=C.0
    BETA=(K-NPT02)*DBETA
    PSI=OMEGA*SPACE*SIN(BETA)/C
  DO 1 I=1,NE
    FPSI=(I-NEC2)*PSI
  DO 1 J=1,NNJ
    FJMI=J-1
    ARGU=-FPSI-FJMI*PHI
1   AMPL=AMPL+W(I,J)*CCS(ARGU)
    AMPLMG=ABS(AMPL)
    XP2(K)=AMPLMG*SIN(BETA)
    YP2(K)=AMPLMG*COS(BETA)
2   CONTINUE
    DO 3 KK=1,NPT
      XP2(KK)=XP2(KK)/AMPMAX
      YP2(KK)=YP2(KK)/AMPMAX
3   CONTINUE
    CALL RESET(IPL0T3) XP2(1),0.0)
    CALL STPOS(IPL0T3,XP2,YP2,NULL,1,NPT)
    CALL PLINE(IPL0T3,0.0,0.0)
    CALL STPOS(IPL0T3,0.0,0.0)
    CALL PLINE(IPL0T3,XP2,YP2,NULL,1,LP)
    CALL EXEC(IPL0T3)
    ICODE2=27
    RETURN 1
  ENTRY PPATRN(*)

```

ENTRY PPATRN GENERATES THE DATA FOR THE  
'DIRECTIONALITY PATTERN' CALCCOMP PLOT.

```

-----
CALL MLTTS(IATL,2)
IT3(10)=IT5(NF1)
IT3(12)=T1
JXY(3)=3
JXY(4)=4
DO 8 I=1,NPT
  XY(I,1)=XP2(I)
  XY(I,2)=YP2(I)
  IF(I.GT.LP)GC TO 7
  XY(I,3)=XXP(I)
  XY(I,4)=YYP(I)
GO TO 8
7 XY(I,3)=XXP(LP)
  XY(I,4)=YYP(LP)
8 CONTINUE
CALL VPLOT(XY,JXY,NPT,256,2,1,-1.1,1.1,0.0,1.616)
WRITE(6,20)NE,FO,BW,DT
20 FORMAT(2I10,3E10,3)
CALL DRAW(NPT,XP2,YP2,1,0,LAB,IT3,0.4,0.4,0.3,2,2,7,5,
*0, LAST)
CALL DRAW(LP,XXP,YYP,3,0,LAB,IT3,0.4,0.4,0.3,2,2,7,5,
*0, LAST)
RETURN 1
END

```

```

SUBROUTINE SFRT(A,NCPLX,ISIGN)
FAST FOURIER TRANSFORM SUBROUTINE FOR 2**N POINTS, (N INTEGER) BY N. BRENNER
PARAMETERS:
A(I)=COMPLEX ARRAY OF NCPLX POINTS WITH: REAL A(I) IN CELL A(2*I-1)
IMAG A(I) IN CELL A(2*I)
NCPLX=NUMBER OF COMPLEX PCINTS=2**N WHERE N IS A POSITIVE INTEGER
ISIGN=NUMBER GIVING DIRECTION OF TRANSFORM: IF ISIGN=-1 PERFORM INVERSE TRANSFORM. IF ISIGN=1 PERFORM FORWARD TRANSFORM FOLLOWED BY INVERSE TRANSFORM. RESULTS IN THE ORIGINAL FUNCTION A(1)
DIMENSION A(1)
N=2*NCPLX
J=1
DO 11 I=1,N,2
  JP1=J+1
  IP1=I+1
  IF(I-J)2,4,4
2 TR=A(J)

```

```

001
002
003
004
005
006
007
008

```

```

TI=A(JPI)
A(J)=A(I)
A(JPI)=A(IPI)
A(I)=TR
A(IPI)=TI
4 M=N/2
5 IF(M-J)6,11,11
6 J=J-M
M=M/2
11 IF(M-2)11,5,5
J=J+M
13 MMAX=X-N)14,99,99
14 IF(MMAX=2*MMAX
INCR=2
PIMI=3.14159265/FLCAT(MMAX)
DO 21 M=1,MMAX,2
ANG=PIMI*FLOAT(M-1)
WR=COS(ANG)
WI=SIN(ANG)
IF(ISIGN)18,99,19
18 WI=-WI
19 DO 21 I=M,N,INCR
J=I+MMAX
JPI=J+1
IPI=I+1
TR=WR*A(J)-WI*A(JPI)
TI=WR*A(JPI)+WI*A(J)
A(J)=A(I)-TR
A(JPI)=A(IPI)-TI
A(I)=A(I)+TR
A(IPI)=A(IPI)+TI
21 A(MMAX)=INCR
GO TO 13
99 RETURN
END

```

```

009
0010
0011
0012
0013
0014
0015
0016
0017
0018
0019
0020
0021
0022
0023
0024
0025
0026
0027
0028
0029
0030
0031
0032
0033
0034
0035
0036
0037
0038
0039
0040
0041
0042
0043

```

```

C VPLOT SUBROUTINE VFLOT(XY,JXY,N,NDIM,ACUR,ISCALE,XL,XU,YL,YU)
DIMENSION IGRID(101),XS(11),YS(11),ICHAR(7),XY(1),JXY(1)
DATA ICHAR/1H+,1H*,1H~,1H$,1H=,1H.,1H /
XMIN=XL
XMAX=XU
YMIN=YL
YMAX=YU
IF(ISCALE.NE.0)GC TO 32
XMAX=-1.0E 20
XMIN=-XMAX
YMAX=XMAX
YMIN=XMIN
J2=0
DO 31 J=1,NCUR
J2=J2+2
J1X=(JXY(J2-1)-1)*NDIM
J1Y=(JXY(J2)-1)*NDIM
DO 31 I=1,N
IJX=J1X+I
IJY=J1Y+I
IF(XY(IJX).GT.XMAX)XMAX=XY(IJX)
IF(XY(IJX).LT.XMIN)XMIN=XY(IJX)
IF(XY(IJY).GT.YMAX)YMAX=XY(IJY)
IF(XY(IJY).LT.YMIN)YMIN=XY(IJY)
31 CONTINUE
XR=XMAX-XMIN
IF(XR.EQ.0.0)XR=1.0E-20
YR=YMAX-YMIN
IF(YR.EQ.0.0)YR=1.0E-20
XT=XMAX*XMIN
YT=YMIN*YMAX
IF(XT.LT.0.0)IYAX=100.0*(-XMIN)/XR+1.5
IF(YT.LE.0.0)IXAX=64.0*YMAX/YR+1.5
XINCR=XR/16.0
YINCR=YR/16.0
XS(1)=XMIN
YS(1)=YMAX
DO 46 I=2,11
XS(I)=XS(I-1)+XINCR
46 DO 47 I=2,17
YS(I)=YS(I-1)-YINCR
47 WRITE(6,10)(XS(I),I=1,11)
II=1
KK=0
DO 146 LINE=1,65
DO 101 J=1,101
101 IGRID(J)=ICHAR(7)

```

000  
001  
002  
003  
004  
005  
006  
007  
008  
009  
010  
011  
012  
013  
014  
015  
016  
017  
018  
019  
020  
021  
022  
023  
024  
025  
026  
027  
028  
029  
030  
031  
032  
033  
034  
035  
036  
037  
038  
039  
040  
041  
042  
043  
044  
045  
046



```

105 IF(YT.GT.0.0)GO TO 109
109 IF(LINE.NE.IXAX)GO TO 109
109 DO 105 J=1,101
109 IGRID(J)=ICCHAR(6)
109 IF(XT.LT.0.0)IGRID(IYAX)=ICCHAR(6)
J2=0
DO 125 J=1,NCUR
JC=MCD(J,5)
J2=J2+2
JIX=(JXY(J2-1)-1)*NDIM
JIY=(JXY(J2)-1)*NDIM
DO 125 I=1,N
IJX=JIY+1
IJY=JIY+I
IPTY=64.C*(YMAX-XY(IJY))/YR+1.5
IF(IPTY.GT.65)IPTY=65
IF(IPTY.LT.1)IPTY=1
IF(IPTY.NE.LINE)GO TO 125
IPTX=10C.0*(XY(IJX)-XMIN)/XR+1.5
IF(IPTX.LT.1)IPTX=1
IF(IPTX.GT.101)IPTX=101
IF(JC.NE.0)GC TO 119
IGRID(IPTX)=ICCHAR(5)
GO TO 125
IGRID(IPTX)=ICCHAR(JC)
119 CONTINUE
125 IF(KK.GT.0)GO TO 134
WRITE(6,20)YS(II),(IGRID(I),I=1,101),YS(II)
II=II+1
GO TO 135
134 WRITE(6,30)(IGRID(I),I=1,101)
135 KK=KK+1
IF(KK.NE.4)GO TO 146
KK=0
146 CONTINUE
WRITE(6,40)(XS(I),I=1,11)
RETURN
10 FORMAT(1H1,1PE15.2,10E10.2/10X,1H*,20(5H+*****),2H**)
20 FORMAT(1PE10.2,1H+,101A1,1H+,E9.2)
30 FORMAT(10X,1H*,101A1,1H*)
40 FORMAT(10X,1H*,20(5H+*****),2H+*/1PE16.2,10E10.2)
END

```

## BIBLIOGRAPHY

1. Tolstoy, I. and Clay, C.S., Ocean Acoustics, McGraw-Hill Book Co., New York, 1966, p.15.
2. Lamb, H. J., Hydrodynamics, Cambridge University Press, 6th ed., 1932, pp. 258, 259.
3. Collin, R. E., Field Theory of Guided Waves, McGraw-Hill Book Co., New York, N.Y., 1960, p.22.
4. Richtmyer, R. D. and Morton, K. W., Difference Methods for Initial Value Problems, Interscience Publishers, New York, N. Y., 1967, pp. 24-25.
5. Brilloun, L., Wave Propagation in Periodic Structures, McGraw-Hill Book Co., New York, 1946.
6. Lighthill, M. J., Fourier Analysis and Generalized Functions, Cambridge University Press, 1959.
7. Crank, J. and Nicholson, P., A Practical Method For Numerical Integration of Solutions of Partial Differential Equations of Heat Conduction Type, Proc. Cambridge Philos. Soc., vol. 43, p. 50.
8. Hadamard, J., Lectures on Cauchy's Problem in Linear Partial Differential Equations, Yale University Press, 1923.
9. Salvadori, M and Baron, M., Numerical Methods in Engineering, Prentice-Hall, Inc., Englewood Cliffs, N. J., chapt. III.
10. Courant, R., Friedrichs, K. O. and Lewy, H., Über Die Partiellen Differenzengleichungen der Mathematischen Physik, Mathematischen Annalen, vol. 100, p. 32, 1928.
11. Kreiss, H. O., Über die Stabilitätsdefinition für Differenzengleichungen die Partielle Differentialgleichungen Approximieren, Nordisk Tidskr. Informations-Behandling, vol. 2, 1962, p. 153.
12. Kreiss, H. O., Über implizite Differenzenmethoden für partielle Differentialgleichungen, Numer. Math., vol. 5, p. 24.
13. Lax, P. D. and Wendroff, B., On the Stability of Difference Schemes, Comm. Pure Appl. Math., vol. 15, 1964, p. 363.
14. Johansson, O., and Kreiss, H. O., Über das Verfahren der zentralen Differenzen zur Lösung des Cauchy-Problems für partielle Differentialgleichungen, Nordisk Tidskr, Informations-Behandling, vol. 3, 1963, p. 97.

15. Kato, T., Estimation of Iterated Matrices With Application to the Von Neumann Condition, Numer. Math., vol. 2, 1960, p. 22.
16. Bagrinovskii, K. A. and Godunov, S. K., Difference Schemes for Multi-dimensional Problems, Dokl. Akad. Nauk. USSR, vol. 115, 1957, p. 431.
17. Yanenko, N. N., On Weak Approximation of Systems of Differential Equations, Sibirsk. Mat. Zh., vol. 5, 1964, p. 1430.
18. Douglas, J. and Gunn, J., A General Formulating of Alternating Direction Methods, I., Numer. Math., vol. 6, 1964, p. 428.
19. Cooley, J. W. and Tukey, J. W., An Algorithm for the Machine Calculation of Complex Fourier Series, Math. of Comp., vol. 19, no. 90, 1965, pp. 297-301.
20. Frazer, R., Duncan, W., and Collar, A., Introduction to Elementary Matrices, Cambridge University Press, 1955, p. 138.
21. Courant, R. and Hilbert, D., Methods of Mathematical Physics, Vol. I, New York, Interscience Publishers, Inc., 1962.
22. Szego, G., Orthogonal Polynomials, American Mathematical Society, New York, 1959.
23. Kelvin, Lord, Popular Lectures, vol. I, 1881, p. 185.
24. Watkins, D. A., Theory of Microwave Devices, John Wiley and Sons, Inc., 1958, p. 2.
25. Schelkunoff, S. A., Communications on Pure and Applied Mathematics, D. van Nostrand, 1943, p. 208, vol. 4, p. 117, 1951.
26. Gentleman, W. M. and Sande, G., "Fast Fourier Transforms - For Fun and Profit," IEEE Transactions on Audio and Electroacoustics, June 1967.
27. Stockham, T. G., "High Speed Convolution and Correlation," AFIPS, vol. 28, 1966 Spring Joint Computer Conference, Spartan Books, Washington, 1966.
28. Johnson, R. M., "Graphical Fourier Operations," Research Report no. 31, U. S. Naval Postgraduate School, October 1962.
29. Brenner, N., "Three Fast Fourier Transform Fortran Programs," Lincoln Lab. Technical Report, no. 67-167, M. I. T.

30. Middleton, D. and Groginski, H., "Detection of Random Acoustic Signals by Receiver with Distributed Elements," J. Acoust. Soc. Am., vol. 38, no. 5, November 1965.
31. Capon, J., Greenfield, R. J. and Kolker, R. J., "Multidimensional Maximum-Likelihood Processing of a Large Aperature Seismic Array," Proc. IEEE, 55 (2), 1967.
32. Cox, LCDR, H., USN, "Interrelated Problems in Estimation and Detection I," Proc. Nato Advanced Study Institute on Signal Processing with Emphasis on Underwater Acoustics, August 1968, Enschede, The Netherlands.
33. Burg, J. P., "Three-Dimensional Filtering with an Array of Seismometers," Geophysics, vol. 29, October 1964, pp. 693-713.
34. Claerbout, J. F., "Detection of P Waves from Weak Sources at Great Distances," Geophysics, vol. 29, April 1964, pp. 197-211.
35. Bekey, G. A. and Karplus, W. J., Hybrid Computation, John Wiley and Sons, New York, 1968.
36. Von Neumann, J., Mathematische Grundlogten der Quantumechanik, Dover, 1943.
37. Hildebrand, F. B., "On the Convergence of Numerical Solutions of the Heat Flow Equation," J. Math. Physics, vol. 31, p. 35.
38. Ramo, S., Whinnery, J. R., and Van Duzer, T., Field and Waves in Communication Electronics, John Wiley and Sons, New York, 1965, pp. 29-30
39. Somerfeld, A., Partial Differential Equations, Academic Press, New York, 1949, p. 41.
40. Widrow, B., "Adaptive Antenna Systems," Proc. of the IEEE, vol. 55, no. 12, December 1967.
41. Mantey, P. E., "Convergent Automatic-Synthesis Procedures for Sampled-Data Networks with Feedback," Tech. Report no. 6773-1, Stanford Electronics Laboratories, October 1964.
42. Whittle, P., Prediction and Regulation by Linear Least-Square Methods, Van Nostrand, Princeton, N. J., 1963.
43. Claerbout, J. F., "Digital Filters and Applications to Seismic Detection and Discrimination," Rept. no. 5, Contract AF19(604)7378 M. I. T., Cambridge, Mass., February 1963.



44. Ragazzini, J. R. and Franklin, G. F., Sampled-Data Control Systems, McGraw-Hill Book Co., New York, 1947.
45. Kron, G., Tensor Analysis of Networks, John Wiley and Sons, New York, 1942.
46. Kalman, R.E., "A New Approach to Linear Filtering and Prediction Problems," Trans, ASME, J. Basic Engineering, March 1960.
47. Lee, R. C. K., Optimal Estimation, Identification and Control, Research Monography No. 28, M. I. T. Press, Cambridge, Mass., 1964.



# INITIAL DISTRIBUTION LIST

|                                                                                                                                           | No. Copies |
|-------------------------------------------------------------------------------------------------------------------------------------------|------------|
| 1. Defense Documentation Center<br>Cameron Station<br>Alexandria, Virginia<br>Attn: IRS                                                   | 20         |
| 2. Assoc. Prof. H. A. Titus, Code 52Ts<br>Department of Electrical Engineering<br>Naval Postgraduate School<br>Monterey, California 93940 | 7          |
| 3. Dr. Roy M. Johnson<br>Department of Electrical Engineering<br>Montana State University<br>Bozeman, Montana 59715                       | 7          |
| 4. Professor W. P. Cunningham, Code 61Cm<br>Department of Physics<br>Naval Postgraduate School<br>Monterey, California 93940              | 1          |
| 5. Library, Code 0212<br>Naval Postgraduate School<br>Monterey, California 93940                                                          | 2          |
| 6. Naval Ship Systems Command<br>Navy Department<br>Washington, D.C. 20360<br>Attn: Dave Pope, Code 2126 MUN                              | 2          |
| 7. Dr. Jerry Kenneson<br>Naval Electronics Laboratory Center<br>San Diego, California 92152                                               | 2          |
| 8. Mr. Dick Reins<br>Fleet Numerical Weather Central<br>Naval Postgraduate School<br>Monterey, California 93940                           | 1          |
| 9. Professor C. E. Menneken, Code 023<br>Dean of Research Administration<br>Naval Postgraduate School<br>Monterey, California 93940       | 2          |

## DOCUMENT CONTROL DATA - R &amp; D

(Security classification of title, body of abstract and indexing annotation must be entered when the overall report is classified)

1. ORIGINATING ACTIVITY (Corporate author)

Naval Postgraduate School  
Monterey, California 93940

2a. REPORT SECURITY CLASSIFICATION

Unclassified

2b. GROUP

3. REPORT TITLE

Large-Domain Discrete Integration Techniques for the Wave Equation as an Aid In the Calculation of Propagation Loss and the Study of Adaptive Acoustic Arrays

4. DESCRIPTIVE NOTES (Type of report and, inclusive dates)

Doctors Thesis, October 1969

5. AUTHOR(S) (First name, middle initial, last name)

Roy Martin Johnson

6. REPORT DATE

October 1969

7a. TOTAL NO. OF PAGES

313

7b. NO. OF REFS

47

8a. CONTRACT OR GRANT NO.

b. PROJECT NO.

c.

d.

9a. ORIGINATOR'S REPORT NUMBER(S)

1

9b. OTHER REPORT NO(S) (Any other numbers that may be assigned this report)

10. DISTRIBUTION STATEMENT

This document has been approved for public release and sale; its distribution is unlimited.

11. SUPPLEMENTARY NOTES

12. SPONSORING MILITARY ACTIVITY

Naval Postgraduate School  
Monterey, California 93940

13. ABSTRACT

This work is concerned with two inter-related problems: (1) the theory and application of algorithms for the discrete integration of the acoustic wave equation in large, multi-dimensional, variable-parameter domains and, (2) the automatic synthesis of adaptive acoustic arrays for the detection and estimation of received acoustic signals which are incident from such domains.

It is shown that the steady-state integration of large domains may be partitioned into subdomain integrations for slowly-varying variable-parameter domains. A new method of integration based upon the Fast Fourier Transform is given. A new method for obtaining closed form coefficient expressions for the Fast Fourier Transform is shown and illustrated.

The results of a general computer-display simulation program for the Widrow feed-forward algorithm are given. Several limitations and possible modifications to the Widrow procedure are given. The use of Kalman-type filters as an alternative method is introduced.

### KEY WORDS

## Acoustic arrays

LINK A

LINK B

LINK C

ROLE

WT

| NAME                | ROLE            |
|---------------------|-----------------|
| Mr. J. Edgar Hoover | Director        |
| Mr. Clegg           | Chief of Bureau |
| Mr. Glavin          | Chief of Bureau |
| Mr. Ladd            | Chief of Bureau |
| Mr. Nichols         | Chief of Bureau |
| Mr. Rosen           | Chief of Bureau |
| Mr. Tracy           | Chief of Bureau |
| Mr. Carson          | Chief of Bureau |
| Mr. Egan            | Chief of Bureau |
| Mr. Gurnea          | Chief of Bureau |
| Mr. Hendon          | Chief of Bureau |
| Mr. Pennington      | Chief of Bureau |
| Mr. Quinn           | Chief of Bureau |
| Mr. Nease           | Chief of Bureau |
| Mr. Gandy           | Chief of Bureau |

WT

ROLE

WT









thesJ6337

Large-domain discrete integration techni



3 2768 001 02695 8

DUDLEY KNOX LIBRARY